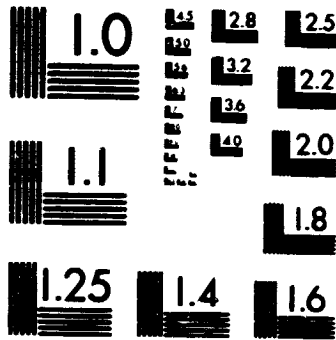


183

6920

3 1/2 B



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

①

AD A120522

Autonomous Spacecraft Project

Autonomous Spacecraft Design and Validation Methodology Handbook

(Issue 1)

Approved for Public Release; Distribution Unlimited

30 April 1982
Interim Report

DTIC
ELECTE
OCT 20 1982
S D H

Prepared for

U.S. Air Force Systems Command
Headquarters, Space Division
Los Angeles, California 90009

Through an agreement with

National Aeronautics and Space Administration

By

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California 91109

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

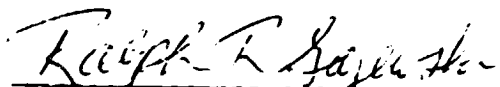
DTIC FILE COPY

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

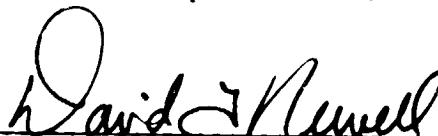
82 10 19 014

This interim report was submitted by the Jet Propulsion Laboratory, California Institute of Technology, 4800 Oak Grove Drive, Pasadena, California, 91109, under Contract NAS7-100, JPL Task Plan No. 80-1487, with the Headquarters, Space Division, Los Angeles AFS, California, 90009. Major Ralph R. Gajewski (YLXS) was the project officer. This report has been reviewed and cleared for open publication and/or public release by the appropriate Public Affairs Office (PAS) in accordance with AFR 190-17 and DODD 5230.9. There is no objection to unlimited distribution of this report to the public at large or by DTIC to the National Technical Information Service (NTIS).

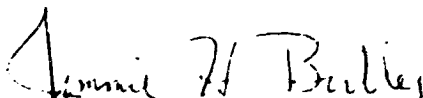
This technical report has been reviewed and is approved for publication.



RALPH R. GAJEWSKI, Major, USAF
Chief, Space Vehicle Subsystems Division
Deputy for Technology



DAVID T. NEWELL, Lt Col, USAF
Deputy Director, Space Systems Planning
Deputy for Technology



JIMMIE H. BUTLER, Colonel, USAF
Director of Space Systems Planning
Deputy for Technology

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER SD-TR-82-58	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Autonomous Spacecraft Design and Validation Methodology Handbook (Issue I)		5. TYPE OF REPORT & PERIOD COVERED Interim
7. AUTHOR(s) Philip R. Turner		6. PERFORMING ORG. REPORT NUMBER 7030-4
9. PERFORMING ORGANIZATION NAME AND ADDRESS Jet Propulsion Laboratory California Institute of Technology 4800 Oak Grove Drive, Pasadena, CA 91109		8. CONTRACT OR GRANT NUMBER(s) NAS7-100 JPL Task Plan No. 80-1487 Rev. A
11. CONTROLLING OFFICE NAME AND ADDRESS HQ Space Division/YLXT Box 92960 Worldway Postal Center Los Angeles, CA 90009		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE 30 April 1982
		13. NUMBER OF PAGES
		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		16. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for Public Release; Distribution Unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Autonomy, autonomous spacecraft, autonomous satellite, fault protection, fault tolerance, on-board computing, autonomous satellite control, autonomous station-keeping, autonomous health and welfare, redundancy management, autonomous navigation, spacecraft system design, spacecraft subsystem design, validation testing, spacecraft design methodology, spacecraft validation methodology		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This Handbook provides reference material pertaining to the design and validation of autonomous spacecraft systems. The goal is to present in one place a compendium of rules, considerations, and approaches to spacecraft autonomy that will be useful to Air Force and Industrial personnel involved in project management, design, implementation, test, and operations of military space systems. The scope of the content ranges from mission level requirements definition to examples of techniques used to implement subsystem level autonomous control.		

DTIC
ELECTED
OCT 20 1982
H

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

BLOCK 20, ABSTRACT (Continued)

An attempt has been made to summarize and document the spacecraft autonomy approaches that JPL utilized in their planetary spacecraft designs over the past decade, and to reorient these approaches to the extent necessary to make them applicable to defense satellites.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

USAF Report SD-TR-82-58
JPL Report 7030-4, Issue 1
JPL D-188

AUTONOMOUS SPACECRAFT DESIGN AND VALIDATION METHODOLOGY HANDBOOK

Prepared by

Philip R. Turner, Design Methodology Task Leader
Autonomous Spacecraft Program

30 April 1982

Approved:



David D. Evans
Manager, Autonomous
Spacecraft Program

Approved:



H. W. Norris
Manager, Air Force
Space Program

Prepared for
U. S. Air Force Systems Command
Headquarters, Space Division
Through an agreement with
National Aeronautics and Space Administration
by

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California 91109

PREFACE

This document was prepared for the USAF Space Division Deputy for Technology (YL) by personnel of the Jet Propulsion Laboratory (JPL). An attempt has been made to summarize and document the spacecraft autonomy approaches that JPL utilized in their planetary spacecraft designs over the past decade, and to reorient these approaches to the extent necessary to make them applicable to military satellites.

The reader will find the resulting material to be a useful collection of autonomy definitions, goals, approaches, guidelines, examples, and lessons learned. Since autonomy practice does not lend itself to display in the form of tabular data and curves, one will not find traditional Mil-Spec handbook-type data.

This Issue I version is being distributed for the immediate use of the community as well as for review and comment from potential users. The current expectation is to revise and update this material during FY'83, hence it is important that users provide comments and suggestions to Space Division (Attn: YLXS), or to the Jet Propulsion Laboratory (Attn: Manager, Autonomous Spacecraft Program, 180-202) as soon as possible.

Accession For	
DTIC TAB	<input checked="checked" type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	<input type="checkbox"/>
By	
Distribution/	
Availability Codes	
Dist	Avail and/or
A	Special
DTIC COPY INSPECTED 2	

TABLE OF CONTENTS

	<u>Page</u>
Part I	
Introduction.	I-14
1. Scope and Purpose.	I-14
2. Constraints and Assumptions.	I-15
3. Handbook Description and Use	I-16
4. Autonomy Issues.	I-19
 Part II	
Generation of Autonomy Specifications	
1. Autonomy as a Top Level Requirement.	II-3
1.1 Need for Top Level Requirements	II-3
1.2 Benefits of Autonomy to a Mission	II-4
1.3 Autonomy in the Space System Life Cycle	II-4
1.4 Autonomous Spacecraft Operations and Ground System Interfaces	II-7
2. Development of Space System Autonomy Specifications	II-10
2.1 Requirements Formulation.	II-10
2.2 Generic Policy Goals.	II-11
2.3 Specification Content	II-14
3. Development of Space Vehicle Autonomy Specifications	II-16
3.1 Requirements Formulation.	II-16
3.2 Generic Implementation Goals.	II-16
3.3 Specification Content	II-22
 Part III	
Design Methodology	
1. The Design Process	III-8
1.1 Spacecraft System Level Autonomous Design	III-8
1.2 Critical Issues in Design	III-18
1.3 Autonomy Implications for the System Development Process	III-27
2. Spacecraft System Autonomous Design Techniques	III-33
2.1 System Level Architecture Techniques and Issues.	III-33

	<u>Page</u>
2.2 Autonomous Navigation	III-66
2.3 Design Requirements for Support of Validation and Testing.	III-112
3. Subsystem Level Autonomous Design Characteristics.	III-117
3.1 Tracking, Telemetry, and Command (TT&C) Subsystem	III-117
3.2 Power Subsystem	III-138
3.3 Attitude Control Subsystem.	III-145
3.4 Propulsion Subsystem.	III-164
3.5 Thermal Control Subsystem	III-173

Part IV

Validation Methodology

1. Validation Philosophies.	IV-2
2. The Validation Process	IV-3
3. Validation Requirements.	IV-3
3.1 Audit Trail Baseline.	IV-3
3.2 Subsystem and System Interfaces	IV-3
3.3 Electromagnetic Interference and Other Transient Phenomena	IV-5
3.4 Diagnostic and Test Software.	IV-5
3.5 Mission Unattended Lifetime Demonstration	IV-6
4. Implementation Methods and Techniques.	IV-6
4.1 The Validation/Test Subsystem	IV-6
4.2 Requirements Evaluation and Traceability.	IV-8

Part V

Summary of Experience with Autonomous Design and Validation

1. Background	V-3
2. Project Level Details.	V-4
2.1 Voyager Project Policies.	V-4
2.2 Galileo Project Policies.	V-4
3. Design Implementation Experience	V-6
4. Validation Experience with Autonomy.	V-8
4.1 Background.	V-8
4.2 Key Philosophies.	V-10
4.3 Mission Validation Objectives and Priorities	V-11

	<u>Page</u>
4.4 Qualification and Flight Acceptance Testing	V-11
4.5 Levels of Testing	V-12
4.6 Engineering Tests	V-12
4.7 Flight Equipment Operating Time	V-13
4.8 Regression Testing.	V-13
4.9 System Tests.	V-13
4.10 Subsystem Tests	V-14
4.11 Post-Environmental Inspection	V-15
4.12 Test Plans and Related Documents.	V-15
4.13 Test Facilities	V-16
5. Autonomy Implementation Recommendations . .	V-17

Appendix A	Viking Orbiter and Voyager Spacecraft Design Summaries
Appendix B	Generalized Algorithms for Autonomous Control and Fault Management
Appendix C	Example Flight Algorithms for Autonomcus Control and Fault Management
Appendix D	Data Processing Architectures and Fault Tolerance Specification

GLOSSARY

ABBREVIATIONS AND ACRONYMS

AACS	-	Attitude and Articulation Control Subsystem
AAI	-	All-Axis Inertial
ACE	-	Attitude Control Electronics
ACS	-	Attitude Control Subsystems
ADS	-	Analog-to-Digital Converter
ADET	-	Angle Detector
AF	-	Air Force
APL	-	A Programming Language
APSM	-	Automated Power System Management
ARIÉS	-	Automated Reliability Interactive Estimate System
ASP	-	Autonomous Spacecraft Project
ATPC	-	Attitude, Translation and Pointing Control
ATPS	-	Attitude, Translation and Pointing Subsystem
bph	-	bits per hour
bps	-	bits per second
CC	-	Celestial Cruise
CCD	-	Charge-Coupled Device
CDR	-	Critical Design Review
CCS	-	Computer Command Subsystem
CDU	-	Command Distribution Unit
CMOS	-	Complementary Metal Oxide Semiconductor
CMS	-	Common Memory Subsystem
CP	-	Central Processor
CPU	-	Central Processing Unit; Command Processing Unit

ABBREVIATIONS AND ACRONYMS (Cont'd)

CST	-	Canopus Star Tracker
DMA	-	Data Memory Access
DMS	-	Data Memory Subsystem
DN	-	Data Number
DPU	-	Distributed Processing Unit
DRIRU	-	Dry Inertial Reference Unit
DSCS	-	Defense Satellite Communications System
DSN	-	Deep Space Network
DSS	-	Data Storage Subsystem
DTR	-	Digital Tape Recorder
EMI	-	Electromagnetic Interference
EOF	-	End of File
EOT	-	End of Table
ESA	-	Electrically Steerable Antenna
ESS	-	Emplaced Science Station
FCP	-	Flight Control Programmer
FDS	-	Flight Data Subsystem
FFT	-	Fast Fourier Transform
FMEA	-	Failure Modes and Effects Analysis
FOV	-	Field of View
GPS	-	Global Positioning System
HGA	-	High Gain Antenna
HRAM	-	Hardened Random Access Memory
HT	-	Heat Transfer

ABBREVIATIONS AND ACRONYMS (Cont'd)

HYBIC	-	Hybrid Interface Circuit
HYPACE	-	Hybrid Programmable Attitude Control Electronics
ICD	-	Interface Control Document
IDET	-	Illumination Detector; Intensity Detector
IDMS	-	Interrupt-Driven Microcomputer Subsystem
I/O	-	Input/Output
IR	-	Infrared
IRAS	-	Infrared Astronomy Satellite
IRIS	-	Infrared Interferometer Spectrometer
IRP	-	Interrupt Processor
IRU	-	Inertial Reference Unit
JPL	-	Jet Propulsion Laboratory
kips	-	kilobits per second
LAD	-	Latest Available Data
LFM	-	Load Fault Management
LGA	-	Lean Gain Antenna
LSI	-	Large Scale Integration
MADAN	-	Multimission Autonomous Navigation and Attitude Reference
MDS	-	Modulation/Demodulation Subsystem
MLI	-	Multi-Layer Insulation
MM	-	Mission Module
MOI	-	Mars Orbit Injection
MTIF	-	Mission Time Improvement Factor
MTTF	-	Mean Time to Failure
n/a	-	Not Applicable

ABBREVIATIONS AND ACRONYMS (Cont'd)

NAV	-	Navigation
NVM	-	Non-volatile Memory
OD	-	Orbit Determination
PCM	-	Phase Change Material
PDR	-	Preliminary Design Review
PPH	-	Pulses Per Hour
PPS	-	Power/Pyro Subsystem
PWRS	-	Power Subsystem
P/Y	-	Pitch/Yaw
RAM	-	Random Access Memory
RCA	-	Reaction Control Assembly
RFI	-	Radio Frequency Interface
RFP	-	Request for Proposal
RFS	-	Radio Frequency Subsystem
RI	-	Roll Inertial
RMS	-	Redundancy Management Subsystem
ROM	-	Read Only Memory
RRS	-	Relay Radio Subsystem
RSS	-	Root Sum Square
RTG	-	Radioisotope Thermoelectric Generator
SAR	-	Synthetic Aperture Radar
S/C	-	Spacecraft
SEC	-	Solar Energy Controller
SIRTF	-	Shuttle Infrared Telescopic Facility
SRIF	-	Square Root Information Filter
S/S	-	Subsystem

ABBREVIATIONS AND ACRONYMS (Cont'd)

SS	-	Subsystem
TBD	-	To Be Determined
TBS	-	To Be Supplied
TCAPU	-	Trajectory Correction and Attitude Propulsion Unit
TCM	-	Trajectory Correction Maneuver
TCS	-	Temperature Control Subsystem
TCT	-	Technical Coordination Team
TDL	-	Telecommunications Development Laboratory
TLM	-	Telemetry
TMU	-	Telemetry Modulation Unit
TT&C	-	Telemetry, Tracking and Command
TWTA	-	Traveling Wave Tube Amplifier
USO	-	Ultra-Stable Oscillator
VCO	-	Voltage Controlled Oscillator
VIS	-	Visual Imaging System
VLC	-	Viking Lander Capsule
XMTR	-	Transmitter

Definitions:

- Algorithm** - The description in step-by-step detail of the logical process required to perform a function.
- Attribute** - A system design characteristic that may be supported by system and subsystem level architecture and technical implementation techniques. Common system attributes are survivability, reliability, maintainability, etc.
- Automatic Process** - A process that is controlled in repetitive fashion until disturbed or modified by external inputs.
- Autonomy** - The ability of a spacecraft to meet mission performance requirements without human intervention or ground support for a specified period of time. This may include routine operations as well as the re-establishment of normal operations after the occurrence of pre-defined faults.
- Autonomous Process** - A process that incorporates control structure logic to assess the appropriateness of its automatic functions from internal and/or external sensory inputs and modify the automatic processes as needed.
- Control Structure** - A series of three actions used to implement control of a process. The actions are: 1) sense the state of an internal or external quantity, 2) direct the initiation of an appropriate response by the system, and 3) act to implement the response.
- Endurability** - The ability of a system to maintain a required level of performance throughout the conflict spectrum.
- Failure** - A fault requiring hardware repair, replacement, or software reprogramming for correction.
- Fault** - A fault is a disruption of the specific logical behavior of a system. The disruption may be transient in nature or persist after occurrence.
- Ground Segment** - The portion of a total mission system implemented by ground, sea-borne, or air-borne resources. Ground segment resources may operated in fixed mode, mobile mode, or both.
- Health** - The operation integrity of a system as affected by faults.

Level	-	This term is used in two separate senses in this document: (1) The level of autonomy of a spacecraft is defined in JPL Document 7030-1 on a scale of 0 to 10. Autonomy will always be mentioned with the word level when it is used in this context. (2) The hierarchical structure of the system development process is referred to as a series of levels.
Methodology	-	A body of methods, rules, and postulates employed by a discipline as well as the analysis of the principles or procedures of inquiry of that discipline.
Mission Phase	-	A portion of mission characterized by fulfillment of a distinct mission goal. Examples are prelaunch, on-orbit initialization and checkout, normal operations, and end-of-life.
Navigation	-	Knowledge and control of the spacecraft trajectory. A navigation system is responsible for satisfying mission requirements for orbit determination, trajectory propagation, and maneuver planning. The first two functions are necessary for ephemeris maintenance. Stationkeeping or orbit maintenance may also require these to support the generation of required maneuver plans and parameters. Navigation may utilize trajectory knowledge to predict related events such as eclipses or occultations.
Operating Mode	-	A specific hardware/software configuration of a spacecraft system or subsystem. A spacecraft will typically have an operating mode for performing orbit adjust velocity changes that is separate from other payload related operating modes.
Routine	-	The software implementation of an algorithm or one of its logical parts.
Space Segment	-	The portion of a total mission system implemented by space-based resources.
Validation	-	A set of analytical, operational test, and simulation procedures by which hardware and software functions, performance, and interfaces are evaluated for compliance with mission and system design requirements.
Welfare	-	The operational integrity of a system maintained by nominal periodic control actions, such as calibrations, software updates, etc.

Contents

Part I

	<u>Page</u>
Introduction	I-14
1. Scope and Purpose.	I-14
2. Constraints and Assumptions.	I-15
3. Handbook Description and Use	I-16
4. Autonomy Issues.	I-19
References	I-20

PART I

INTRODUCTION

SECTION 1

SCOPE AND PURPOSE

This Handbook provides reference material pertaining to the design and validation of autonomous spacecraft systems. The goal is to present in one place a compendium of rules, considerations, and approaches to spacecraft autonomy that will be useful to Air Force and Industrial personnel involved in project management, design, implementation, test, and operations of military space systems. The scope of the content ranges from mission level requirements definition to examples of techniques used to implement subsystem level autonomous control.

Autonomy is the ability of a spacecraft to meet mission performance requirements, for a specified period of time, without external support. There are four major functional areas that may be considered as part of spacecraft autonomy. These are:

- (1) Health and Welfare Maintenance
- (2) Navigation
- (3) Command Sequence Generation
- (4) Payload Data Processing

The first two functions are similar in nature for most missions. The last two are highly dependent upon mission and payload requirements. A complex mission may be highly dependent upon ground support of the payload while still being able to benefit from autonomous design for the first two.

This issue of the Handbook will address spacecraft health and welfare maintenance functions and navigation functions. Payload or mission interface issues that do arise are handled in as generic a manner as possible.

SECTION 2

CONSTRAINTS AND ASSUMPTIONS

A set of design and implementation goals for autonomy may be found in "Goals for Air Force Autonomous Spacecraft", SD-TR-81-72, 31 March 1981. The Handbook is consistent with these goals and is intended to provide a tool for implementation of the goals by projects interested in specific mission applications.

The content of this issue of the Handbook is based primarily upon planetary spacecraft project experience. The impact of autonomy upon the design and validation processes has been assessed by examination of existing military spacecraft designs for their current degree of operational autonomy. This impact for the Defense Satellite Communications System (DSCS) III Satellite System is discussed in "Assessment of Autonomous Options for the DSCS III Satellite System," SD-TR-81-87, 6 August 1981. It is found that a high level of spacecraft health and welfare autonomy can be achieved through implementation of existing technology. Advances in technology are required to reduce the overall impact of added autonomy on mass and power. Technology improvements will also have a great impact upon spacecraft control authority architecture.

The application of this Handbook does not depend upon optimizing spacecraft system architecture to support autonomous operation. It does, however, offer information that can be applied to increase the autonomy of an existing design or support the design of a new autonomous spacecraft system.

Faults due to hostile action are not explicitly addressed in the Handbook. The emphasis here is to enhance endurability of the spacecraft in the presence of faults that might occur in normal operation. Some faults caused by hostile action may be properly corrected by the same fault management responses that protect against faults occurring in normal operations.

The material presented in this Handbook is limited in scope to the space vehicle system and subsystem levels of detail. Some references to specific piece parts occur, but subsystems are characterized by their next lowest level subassemblies. No information is included to support design or validation below the subassembly level.

SECTION 3

HANDBOOK DESCRIPTION AND USE

The Handbook is organized into separate parts that discuss specification of autonomy, design methodology, validation methodology, and a summary of important points learned through prior experience and specific implementation approaches. Material that expands upon specific topics or provides supporting rationale is included as appendices.

The material provided in the Handbook covers a wide scope of detail and interest. Figure I-1 presents an organization of the Handbook content by topic of interest (Specification, Design, Validation) and increasing level of detail. Used along with the Table of Contents it should allow the user to locate specific material of interest and obtain a better understanding of the logical organization of the Handbook.

Part II of the Handbook discusses the generation of autonomy specifications. It does not present a "boilerplate" specification example of autonomy. Rather, it presents a rationale for specification of autonomy at all levels of the Space System and addresses specific requirements at Space System Specification and Space Vehicle Specification levels. Each level of specification is addressed with a section of explanation and rationale, a section specifying candidate requirements, and a section identifying portions of the specification format where autonomy requirements are appropriate. The specification formats were derived from References 1 and 2 and are in compliance with MIL-STD-483 and MIL-STD-490. The user is expected to review the sample requirements and select a set that is relevant to his mission needs.

Part III addresses the design methodology for autonomous spacecraft at the spacecraft system and subsystem levels. Section 1 and its subsections address the process of identifying functions for autonomous implementation. A procedure for selection of autonomous functions is presented, along with identification of critical issues and the impact upon program management procedure. Section 2 presents detailed topics on autonomous design at the spacecraft system level. Architecture and control algorithms, navigation subsystem design, and design requirements imposed by validation and test considerations are the principal points of the section. Section 3 addresses the autonomous welfare maintenance and autonomous fault management characteristics of typical spacecraft subsystems.

Part IV characterizes the impact of autonomy on Validation Methodology. The validation process, generation of validation requirements, and specific critical issues are discussed. Some implementation techniques that relate to new requirements generated by autonomy are presented in Section 4.

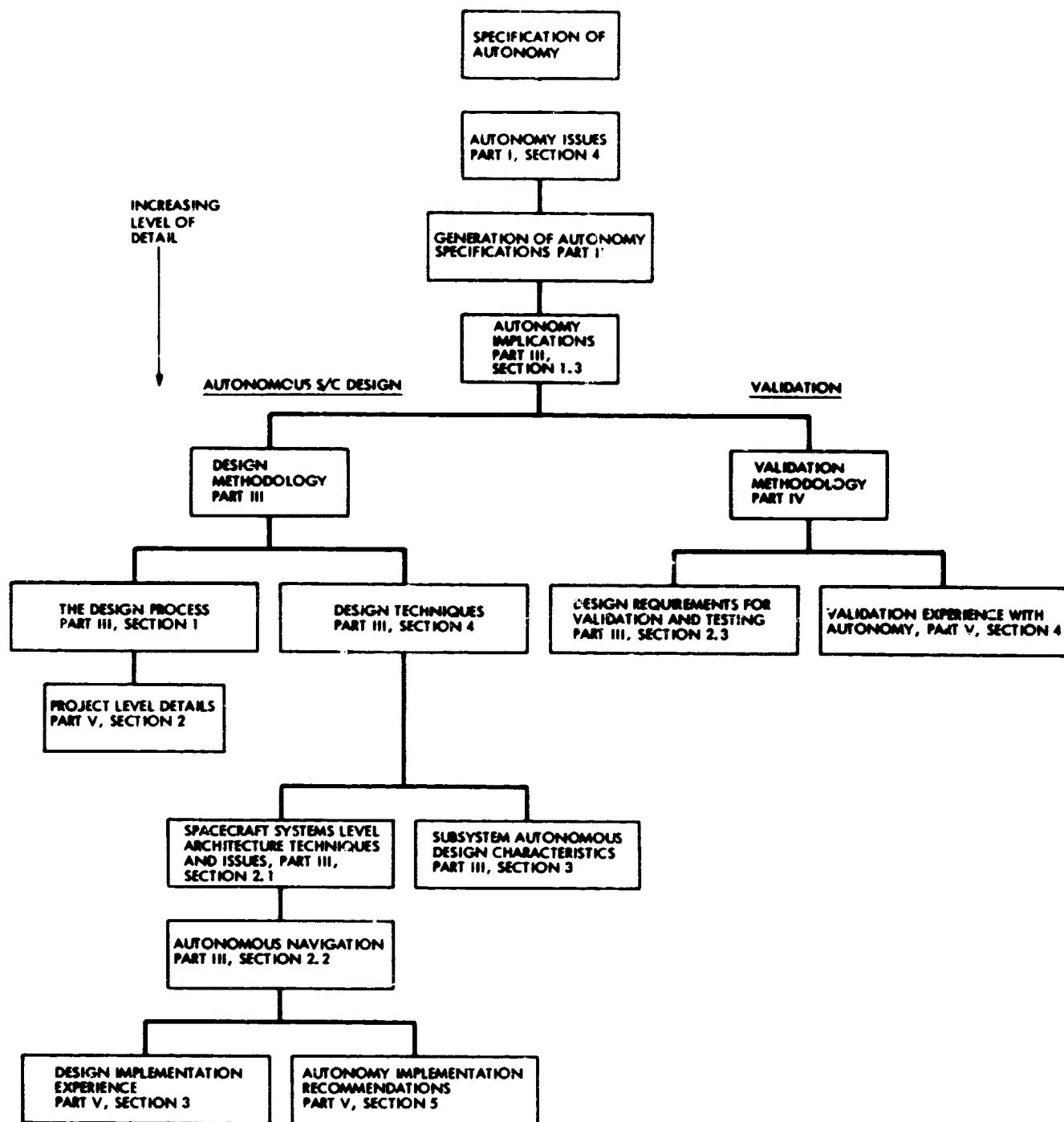


Figure I-1. Handbook Organization by Subject Matter

Part V is intended to condense previous flight project experience with autonomy into a set of topical "lessons". These lessons reflect both positive and negative views of specific design, validation, and operating techniques.

Appendix A contains a discussion of JPL flight experience with autonomy on the Viking Orbiter and Voyager spacecraft. Autonomy implementation is detailed and design details are provided for spacecraft of both programs.

Appendix B contains specific examples of autonomous control and fault management algorithms for generic spacecraft subsystems. The algorithms were generated without regard for application on a specific project.

Appendix C contains selected algorithms actually implemented in the Viking and Voyager flight projects. Requirements and implementation logic are explained in detail, and any experience with operation of the algorithms is noted.

Appendix D contains the study reports resulting from consideration of centralized, decentralized and hybrid architectures for control and data processing capabilities.

SECTION 4

AUTONOMY ISSUES

The implementation of spacecraft autonomy raises a number of issues that should be examined in the context of the requirements of the individual missions. The Handbook provides material that directly relates to these issues or which supports the mission unique analysis needed to resolve these issues. Specific issues of concern are:

- (1) Selection of appropriate goals and requirements for implementation of autonomy. (Part II, Section 1)
- (2) Specification of requirements for design, validation, and implementation of autonomy. (Part II, Sections 2 and 3)
- (3) Impact of autonomy implementation on program management. (Part III, Section 1.2)
- (4) Methodology differences between the implementation of autonomy in new or in existing systems. (Part III, Section 1.2)
- (5) Centralized or decentralized spacecraft control architecture. (Part III, Section 2.1)
- (6) Allocation of autonomous control among systems and subsystems and the degree of executive control required (Part III, Sections 1.1, 2.1, and 3)
- (7) Selection and prioritization of a set of functions as candidates for autonomous operation. (Part III, Section 1.1)
- (8) The degree and nature of ground support required. (Part II, Section 1.4)
- (9) The role of autonomy in reducing operating costs. (Part II, Section 1.2, 1.3)
- (10) The use of redundancy at system and subsystem levels. (Part II, Section 2.1.2)

REFERENCES

1. "Space System Specification Preparation Guide", SD-GB-4, 1 April '81, The Aerospace Corp. for USAF Space Division (SD/AQM).
2. "Space Vehicle Development Specification Preparation Guide", SD-GB-6, 1 April '81, The Aerospace Corporation for USAF Space Division (SD/AQM).

Contents

Part II-Generation of Autonomy Specifications

	<u>Page</u>
1. Autonomy as a Top Level Requirement.	II-3
1.1 Need for Top Level Requirements.	II-3
1.2 Benefits of Autonomy to a Mission.	II-4
1.3 Autonomy in the Space System Life Cycle.	II-4
1.3.1 Concept Definition Phase	II-6
1.3.2 Mission Definition Phase	II-6
1.3.3 Acquisition Phase.	II-6
1.3.4 Operations Phase	II-6
1.4 Autonomous Spacecraft Operations and Ground System Interfaces	II-7
1.4.1 Current Health and Welfare Support	II-7
1.4.1.1 Telemetry Processing	II-7
1.4.1.2 Performance Trend Analysis	II-7
1.4.1.3 Anomaly Investigation.	II-8
1.4.1.4 Welfare Maintenance Control.	II-8
1.4.2 New Ground Functions	II-8
1.4.2.1 Audit Trail Analysis	II-9
1.4.2.2 Software Development Support	II-9
1.4.2.3 Spacecraft Simulation.	II-9
2. Development of Space System Autonomy Specifications.	II-10
2.1 Requirements Formulation	II-10
2.1.1 Scope of Autonomy.	II-10
2.1.2 Duration of Autonomy	II-11
2.2 Generic Policy Goals	II-11
2.2.1 Ground Interaction Reduction	II-12
2.2.2 Spacecraft Integrity Maintenance	II-12
2.2.3 Autonomous Features Transparency	II-13
2.2.4 Onboard Resource Management.	II-13
2.3 Specification Content.	II-14
2.3.1 Section 1. - Scope	II-14
2.3.2 Section 2. - Applicable Documents.	II-14
2.3.3 Section 3. - Requirements.	II-14
2.3.4 Section 4. - Quality Assurance Provisions.	II-15

	<u>Page</u>
3. Development of Space Vehicle Autonomy Specifications	II-16
3.1 Requirements Formulation	II-16
3.2 Generic Implementation Goals	II-16
3.2.1 Systems (Including Thermal Control & Validation)	II-17
3.2.2 Electrical Power and Pyrotechnics.	II-18
3.2.3 Attitude, Translation, and Pointing Control (ATPC)	II-18
3.2.4 Data Processing.	II-19
3.2.5 Payload.	II-19
3.2.6 Telemetry, Tracking, and Command (TT&C)	II-20
3.2.7 Navigation	II-20
3.2.8 Propulsion	II-21
3.3 Specification Content.	II-22
3.3.1 Operational Concepts	II-22
3.3.2 Performance and Physical Characteristics	II-22
3.3.3 Computer Resources	II-22
3.3.4 Subsystem Level Requirements	II-22
3.3.5 Quality Assurance Provisions	II-22

PART II
GENERATION OF AUTONOMY SPECIFICATIONS

SECTION 1
AUTONOMY AS A TOP LEVEL REQUIREMENT

1.1 NEED FOR TOP LEVEL REQUIREMENTS

The potential impact of autonomy on cost, risk, and operations makes it essential to consider the required degree of autonomy at the highest levels of concept definition and mission definition. The impact of autonomy on a program will vary greatly with the degree of autonomy required of the mission and the complexity and cost of achieving all mission requirements. A set of autonomy requirements for a program may be formulated by analyzing a set of generic autonomy goals, selecting a specific subset compatible with the mission, and prioritizing them with other mission requirements.

Several factors influence the importance of this high level selection of autonomy requirements.

- (1) Autonomous operating features will be implemented at much lower levels of the development process, but their design may be undisciplined and unproductive unless guided by higher level requirements.
- (2) A program whose space segment is managed by an agency separate from those responsible for ground control and user segments needs high level visibility of autonomy requirements to ensure compatible implementation and justify increased front end costs.
- (3) The cost impacts of autonomy are more visible when they are directly traceable to major program policies or requirements.
- (4) Potential operational benefits of autonomy may be lost or lessened by ad hoc implementation at low levels of design.
- (5) Autonomy is closely related with other high level attributes that produce major requirements. Specifically, it involves reliability, survivability, endurability, and operability.
- (6) Autonomy is a new attribute with significant impacts on program management and contractual relationships.

1.2 BENEFITS OF AUTONOMY TO A MISSION

A specific mission may have a top level survivability/autonomy requirement, or may benefit from addition of autonomous operating features to complement any or all of its ground related functions. These functions fall into the four classes of health and welfare maintenance, navigation, command sequence generation, and payload data processing. The feasibility and cost/benefit ratio of making these classes autonomous is highly dependent upon the nature of the individual mission. Health and welfare maintenance is the area that has the most common application to all missions and can provide a number of benefits.

- (1) Provide insurance that this spacecraft will perform its mission function in times of critical need.
- (2) Provide increased durability of a spacecraft in the absence of ground support.
- (3) Reduce the need for on-orbit or ground stored spare spacecraft.
- (4) Reduce the impact of ground support loss on user operations.
- (5) Reduce ground support load for normal operations.
- (6) Protect the spacecraft from the consequences of cascading faults.
- (7) Decrease personnel and functional requirements on mobile ground support facilities.
- (8) Provide increased reliability for missions with limited availability or little payload requirement for ground support.
- (9) Provide enhanced telemetry data collection capability for periods when ground support is not available.
- (11) Protect user resources by providing a "fail operational" mode for many categories of faults.
- (11) Simplify anomaly investigation by limiting the impact of faults and recording fault indications and response.

1.3 AUTONOMY IN THE SPACE SYSTEM LIFE CYCLE

System life cycle terminology considered in this Handbook is shown in Table II-1. Activities and major products affected by autonomy are also shown. The life cycle may be related to a new project requiring



autonomy from its inception or to a product improvement block of spacecraft for an existing program. These two separate program circumstances are considered to affect the scope and context of each life cycle phase, but not the basic existence of the phases. The effect of both cases on design methodology is discussed in Topic 1.3.5 of Part III.

1.3.1 Concept Definition Phase

The appropriate time to consider the need for autonomous payload operating features and their impact on autonomous health and welfare and navigation is during the formulation of the initial mission concept, or when changes are made to that concept. The overall mission may not be well enough defined at this stage but the potential benefits of autonomy and associated tradeoffs among space and ground segments could still be identified.

1.3.2 Mission Definition Phase

This phase should result in a firm set of autonomy requirements for the space segment and the space vehicle. Autonomy impacts on other Space Systems Segments should be identified and recorded in the appropriate interface specifications. The Space Segment Specification and Space Vehicle Specification documents are considered the primary means of specifying autonomy requirements to contractors in the acquisition phase. The Request for Proposals (RFP) may define the concepts of autonomy associated with the program if this is not adequately clear from the specifications.

1.3.3 Acquisition Phase

The design and validation methodology is applied to implement autonomous operating functions. Functional requirements and detailed design specifications should document the implementation of autonomy as required by the higher level specifications. Interfaces with components outside the space segment should be well documented. Review and audit activities should address autonomy as a specific issue and provide the mechanism for insuring that implementation of autonomy meets specifications.

1.3.4 Operations Phase

Operations provide the critical final validation experience for the autonomy implementation selected. On-orbit testing and early operational experience allow the assessment of the design of autonomous features. Reprogrammable software implementation and adequate memory and computer performance margins allow for the incorporation of changes to the autonomous features as dictated by flight experience and changes in the vehicle as its operational life advances.

1.4 AUTONOMOUS SPACECRAFT OPERATIONS AND GROUND SYSTEM INTERFACES

Spacecraft autonomy affects the nature and frequency of ground control interfaces. Many ground functions will remain unchanged in nature to support executive over-ride capability while not being frequently required for normal operations. New ground functions will be required to support the onboard processing required for autonomy. The degree of change in ground operations is directly related to the scope of autonomy required. This section will concern itself with autonomy for health and welfare maintenance functions and will not specifically address navigation, command sequence generation, or payload data processing. Effects on ground support of health and welfare maintenance can be classified as changes to existing functions and addition of new functions.

1.4.1 Current Health and Welfare Support

Current functions tend to fall into four categories:

- (1) Telemetry Processing
- (2) Performance Trend Analysis
- (3) Anomaly Investigation
- (4) Repetitious Control for Welfare Maintenance

1.4.1.1 Telemetry Processing. Telemetry data acquisition and processing is not a continuous activity for most missions. Data is typically acquired from the spacecraft at periodic intervals ranging from once or twice per orbit to once per week. The sample of data acquired in real time or recorded since the last readout is processed for analysis by ground support personnel. Autonomous spacecraft design does not interfere with the process in any way. A basic autonomy goal expressed in "Goals for Air Force Autonomous Spacecraft" and Section 2.2 of this document is for the spacecraft autonomy to be transparent to user and ground control activities. The telemetry data acquisition and processing may be aided by some autonomous features. The audit trail requirement for autonomous control actions may include provisions for recording snapshots of telemetry data during periods when ground support is not available. This would allow analysts to have access to data under preselected conditions without the constraint of scheduling support at the specific time. The spacecraft may have the capability for some reprogramming of the telemetry stream contents and format in-flight. Such a feature can be included in autonomous design, making use of basic software programmable processing resources required to implement autonomous control features. In summary, autonomy does not impede the collection of telemetry as desired. If anything, the capability may be enhanced.

1.4.1.2 Performance Trend Analysis. Analysis of performance trends from telemetry data can be carried out in a normal manner. Enhancements to telemetry collection mentioned above may give the analyst additional aid over current spacecraft. Analysis of audit trail data gathered during periods of

autonomous operation may be more meaningful than current "snapshot" practices. An onboard system programmed to report exceptional telemetry events to the audit trail could provide evidence of intermittent faults or operating trends much earlier than would be possible from periodic real time data collections.

1.4.1.3 Anomaly Investigation. Anomaly investigation would be greatly simplified and risk of spacecraft loss would be reduced by autonomous fault management. Ground operations response to an anomaly in a non-autonomous spacecraft is a carefully orchestrated action. Spacecraft safety is the primary concern and no commanding action is taken until data analysis gives the clearest possible understanding of the problem. This process is proper and quite necessary as non-autonomous spacecraft can provide little support to the process. All their actions and responses must be supplied from the ground after the fault has occurred and its consequences have propagated through the spacecraft. An autonomous spacecraft can be programmed to react to specific fault symptoms in the same sense that contingency operations plans are established on the ground for response to serious faults. Control authority on the spacecraft allows rapid corrective action that can prevent the consequences of a fault inducing more serious faults in other spacecraft subsystems. Such cascading faults can easily lead to loss of the spacecraft or a complex and risky recovery process requiring time, expense, and loss of user resources. The primary goal of autonomous fault maintenance is to provide for recovery from a fault or establishment of a safe configuration before serious consequences can occur. Design of the hardware and software algorithms is carried out under a set of rules that consider spacecraft safety as the prerequisite of all autonomous control action. All autonomous control actions and the spacecraft telemetry data relevant to the fault can be stored in the audit trail records to provide ground analysts with the data needed to diagnose the onboard conditions surrounding the fault. The occurrence of a fault and corrective action will be apparent to the ground control segment at the next service contact. Audit trail and additional telemetry data may be analyzed to determine the consequences of the fault, and the ground personnel may act as desired to follow up the autonomous control actions. Simple faults that would have been readily apparent will still be readily apparent to the analysts, and serious faults will have been detected before their consequences mask the original fault.

1.4.1.4 Welfare Maintenance Control. A basic goal of autonomous design is to reduce the requirement for the magnitude and frequency of routine ground activity. Time and resource consuming repetitive actions for calibration, thermal control, spacecraft mode reconfiguration, battery charge maintenance, and a variety of other functions can be performed by use of autonomous control.

1.4.2 New Ground Functions

Additional requirements to support an autonomous spacecraft are:

- (1) Audit Trail Analysis

(2) Software Development Support

(3) Spacecraft Simulation

1.4.2.1 Audit Trail Analysis. The audit trail of the spacecraft contains a record of all autonomous control actions, selected data on the initiation and success of the actions, and spacecraft state changes. Expanded telemetry data or "snapshots" may also be provided. Automated analysis of this data can rapidly provide controllers with the status of the spacecraft and the identification of any fault management activity.

1.4.2.2 Software Development Support. Autonomous control algorithms are largely implemented through onboard software. Ground support will be responsible for the design, coding, verification testing, and uplinking of new or modified software and data tables.

1.4.2.3 Spacecraft Simulation. Some level of spacecraft functional simulation at system or subsystem levels will be required to support software development and anomaly investigation. Careful development of simulation requirements is necessary to assure that enough simulation is supplied to allow for validation of critical functions without involving excess complexity and cost.

SECTION 2

DEVELOPMENT OF SPACE SYSTEM AUTONOMY SPECIFICATIONS

2.1 REQUIREMENTS FORMULATION

The selection of detailed autonomy requirements for the space system is influenced by several factors. The most notable are:

- (1) Mission Payload/User requirements.
- (2) Mission autonomy requirements.
- (3) Constraints due to cost or inheritance from previous design.
- (4) Project Office and Agency policies.

The first three factors will be project unique and are derived from specific concept and mission definition analysis activities. Project Office and agency policies on space system autonomy may be formulated based upon the results of these mission specific analyses and prior experience. The objective of these policies is to guide the selection of space system and space vehicle requirements from a set of generic autonomy goals. Specific policies that aid autonomy requirements definition are:

- (1) Elimination of single point failures.
- (2) Space System and Space Segment Reliability and Endurability.
- (3) Onboard Resource Margins.
- (4) Reprogrammability of Control and Fault Management.
- (5) Scope and Definition of Fault Management.
- (6) Operation in High Level-of-Conflict Environments.

These policies and the mission specific factors are combined to aid the specification of the two most fundamental requirements of autonomy - its scope and duration.

2.1.1 Scope of Autonomy

The scope of autonomy comprises the major space system functions which are to be autonomous and the identification of the external functions of which they are to be autonomous. The basic categories of space system functions have been identified as:

- (1) Health and Welfare Maintenance.
- (2) Navigation.

(3) Command Sequence Generation.

(4) Payload Data Processing.

The space system specification must identify the autonomous functions to be required of the mission and the relation between the autonomous space segment functions and external segments.

In addition to specifying the functional areas affected by autonomy, it is necessary to identify the degree of allowable ground support. Specific points are:

- (1) Type of service support to be supplied at ground contacts.
- (2) Autonomy functions as normal or contingency operation features.
- (3) Identification of functions required to be non-autonomous.
- (4) Nature of ground support to be required for on-orbit testing and anomaly investigation.
- (5) Definition of any reduced performance allowed during autonomous operations.

2.1.2 Duration of Autonomy

The duration required for autonomous operation must be specified to complete the requirement for autonomy. The duration may be specified in different manners for one or all autonomous functions. Potential durations may include:

- (1) Operation with reduced ground support.
- (2) Operation while meeting nominal spacecraft or payload performance requirements.
- (3) Operation with reduced performance requirements.

A set of goals investigated by the Autonomous Spacecraft Program (ASP) were to operate with nominal mission performance for at least 60 days from the last ground contact and to operate with some specifiable degradation of performance for at least six months from the last ground contact.

2.2 GENERIC POLICY GOALS

A set of generic autonomous spacecraft policy goals has been formulated in SD-TR-81-72 to serve as a starting point for the selection of space system level requirements. Four categories of policy goals are identified:

- (1) Ground interaction reduction.
- (2) Spacecraft integrity maintenance.
- (3) Autonomous features transparency.
- (4) Onboard resource management.

An additional set of implementation goals is identified for space vehicle level specification and listed in Section 3 of this Part. Some of these implementation goals are significant enough to be levied as a space system level requirement.

2.2.1 Ground Interaction Reduction

Autonomous spacecraft shall be capable of successfully performing their mission function for an extended period of time, without ground support, and at a specified level of conflict. Specifically:

- (1) Autonomous spacecraft shall operate without performance degradation for at least 60 days from the last initialization update.
- (2) Autonomous spacecraft shall operate for at least six months from the last initialization update. They shall do so within acceptable performance degradation limits for mission-prioritized functions as defined by each mission.
- (3) Autonomous spacecraft shall be able to recover from certain mission-unique failure modes. These failure modes shall be identified and prioritized.
- (4) Autonomous spacecraft shall be capable of restoring themselves to nominal mission performance after occurrence of a combination of non-simultaneous faults, defined a priori, subject to the availability of spare resources. Knowledge of occurrence of such faults shall be available to the ground segment upon request.
- (5) Autonomous spacecraft shall tolerate transient faults without significant loss of mission capability. Knowledge of occurrence of such faults shall be available to the ground segment upon request.

2.2.2 Spacecraft Integrity Maintenance

The integrity of the payload data stream and usefulness of the spacecraft shall not be reduced by the addition of autonomous features. Specifically:

- (1) Autonomous features shall not decrease the performance and functional capability of the spacecraft.
- (2) Autonomous features shall not adversely affect the wearout mechanisms or consumables of the spacecraft.
- (3) Autonomous features shall not appreciably increase the period required for checkout and initialization on-orbit.

2.2.3 Autonomous-Features Transparency

Autonomous features shall be transparent to the spacecraft user. (Exceptions may include periods of fault isolation and recovery following a fault or periods during orbit maneuvering.) Specifically:

- (1) Autonomous spacecraft shall be maintained in a state such that they are capable of receiving ground commands.
- (2) The ground segment shall be able to exert executive control over autonomous management activities of the spacecraft. Faults or combinations of non-simultaneous faults shall not prevent executive control by the ground segment.
- (3) Autonomous maintenance and fault management actions will be designed to operate with minimum impact on operation of the user's payload.

2.2.4 Onboard Resource Management

Management of onboard resources is mission- and mode-dependent. One may choose to accept a shortened useful lifetime in order to obtain maximum performance in a high level-of-conflict situation. Specifically:

- (1) Autonomous spacecraft shall be capable of adjusting space-system performance for various mission-critical modes by managing available spare resources and expendables even in the presence of faults.
- (2) Software that implements autonomous functions shall be reprogrammable from the ground.
- (3) Software shall accommodate reprogramming so that, in the event of depletion of certain resources and/or expendables, mission performance can be maximized within the limitations of the remaining resources.
- (4) Data storage resource, for traceability of autonomous control actions and storage of telemetry data, should be autonomously managed.

2.3 SPECIFICATION CONTENT

This section discusses particular portions of a Space System Specification affected by autonomy. The specification document format and content are described by SD-GB-4, "Space System Specification Preparation Guide", 1 April 1981. The applicability of comments for each section of the specification is dependent upon the autonomy requirements of the program being specified. Section and paragraph numbers referenced in the text refer to sections and paragraphs in the Specification, not in this Handbook.

2.3.1 Section 1. - Scope

There is no specific applicability of autonomy to this section.

2.3.2 Section 2. - Applicable Documents

This section should list all references to autonomy definition or implementation practices that are to be incorporated in the Space System. This Handbook is a representative candidate for inclusion in the specification.

2.3.3 Section 3. - Requirements

(1) Operational and Organizational Concepts - Paragraph 3.1.7.

The subparagraphs on launch concepts and on-orbit operations concepts should contain a description of the autonomous operations for each phase of the mission. On-orbit test requirements for autonomous operations features and the basic concept for autonomous control during the nominal mission should be described.

(2) Characteristics - Section 3.2

The performance characteristics (3.2.1) may describe the system level requirement for spacecraft autonomy in each operational phase and mode (3.2.1.1). The scope and duration of autonomy may be specified for each separate operational phase or mode. Autonomy should be related to the Endurance (3.2.1.3) requirement for the system. Autonomy may support the full system endurance requirement or be limited to certain specific functions. Autonomy requirements that affect physical characteristics (3.2.2), reliability (3.2.3), and maintainability (3.2.4) should be included in the appropriate subparagraph of this section.

(3) Design and Construction - Section 3.3

Computer resource specifications are covered in paragraph 3.3.8. Computer resource margin requirements appear to be adequate for initial program development, but might be tailored for different margins at different phases of the project or for flight versus ground resources.

2.3.4 Section 4. - Quality Assurance Provisions

This section should identify validation requirements or concepts associated with autonomy. Section 4.2 on Quality Conformance Inspections identifies requirements on Design Verification Tests (4.2.2), Qualification Tests (4.2.3), Acceptance Tests (4.2.4), Pre-Launch Validation Tests (4.2.6), Operational Tests (4.2.7), and Independent Validation of Computer Programs.

SECTION 3

DEVELOPMENT OF SPACE VEHICLE AUTONOMY SPECIFICATIONS

3.1 REQUIREMENTS FORMULATION

The space system requirements of Section 2 can be translated to specific space vehicle performance requirements oriented to functions at vehicle system and subsystem levels. A generic set of autonomy implementation goals is analyzed to produce a set of applicable requirements. Factors influencing the selection are similar to those in Section 2.1. In particular, the vehicle requirements should have the following characteristics:

- (1) Compatibility with mission/payload requirements.
- (2) Implement space system autonomy requirements.
- (3) Allow ample freedom for detailed design tradeoffs.
- (4) Describe traceable, verifiable autonomy features.
- (5) Conform to higher level space system design and implementation policies.

3.2 GENERIC IMPLEMENTATION GOALS

Implementation goals should be formulated to provide system and subsystem level goals consistent with the policy goals of Section 2. They provide guidance in the selection of specifications and requirements on autonomous features in spacecraft design. Eight categories can be described to address the major functional areas of spacecraft design.

- (1) Systems (including Thermal Control and Validation)
- (2) Electrical Power and Pyrotechnics
- (3) Attitude, Translation, and Pointing Control (ATPC)
- (4) Data Processing
- (5) Payload
- (6) Telemetry, Tracking, and Command (TT&C)
- (7) Navigation
- (8) Propulsion

The following set of goals was formulated to explicitly characterize a specific level of autonomous capability. Duration of autonomous operations was chosen as 60 days with nominal mission performance and six months with acceptable degraded mission performance. These durations were selected as representative values that were achievable with the current body of experience and design technique, and as being realistic for health and welfare maintenance and navigation functions.

3.2.1 Systems (Including Thermal Control and Validation)

- (1) The hardware and software architectures chosen shall not preclude the ability to add additional autonomous capabilities.
- (2) The system shall be capable of reconfiguration of spare resources at the lowest practical and reasonable level.
- (3) During autonomous operation, performance degradation may be allowable in specific cases, but only after spares are exhausted. "Graceful" degradation is preferred over precipitous change. Where possible, autonomous functions shall mitigate the effects on performance of a functional failure which occurs after spares are exhausted.
- (4) The adverse effects of faults shall not propagate beyond a subsystem interface if the faulty subsystem possesses sufficient spare resources to recover from the fault condition. Ambiguous faults within subsystem interfaces and subsystems' shared resource allocation shall be resolved by system-level mechanisms.
- (5) All fault detection and switching mechanisms shall be designed to minimize false alarms.
- (6) The spacecraft shall manage propellant usage during autonomously conducted orbit-adjust maneuvers (stationkeeping and/or relocation/restoration) to assure that mission lifetime requirements are met.
- (7) The spacecraft shall maintain system temperature control for all functional states and mission thermal environments. Furthermore, the thermal-control function shall autonomously ensure, for all mission phases, that non-catastrophic subsystem failures cannot induce thermal failures which will cause propagation of the initial failure within the satellite system.
- (8) The spacecraft shall utilize selected parametric data (electrical profiles, thermal characteristics, and state changes in the ambient environments, as a minimum) for onboard forecasting of incipient fault conditions within each of the functional areas.
- (9) The execution of any autonomous event or activity not involving fault management shall not be permitted to conflict with other (planned or autonomous) activities.

- (10) The spacecraft shall maintain performance and state-change records (an audit trail) to allow for reconstruction of performance, fault detection, and fault correction activities and determination of the status of resources and expendables.
- (11) The autonomous spacecraft shall maintain the spacecraft center-of-mass and center-of-pressure within limits required to support the mission.
- (12) Spacecraft autonomy shall be capable of being validated on the ground and verified on-orbit.
- (13) Validation shall determine the design margin (when applicable) of the autonomous mechanisms.
- (14) On-orbit verification and testing of autonomy features shall be accomplished without disrupting normal space-segment operations wherever possible. In those cases where some disruption is unavoidable, restoration of normal space-segment operation shall be an entirely autonomous process which is performed in a timely manner.

3.2.2 Electrical Power and Pyrotechnics

- (1) Detection and isolation of load faults in power-bus loads shall be accomplished.
- (2) Power-margin management for power bus (including power sources, power-conditioning elements, and user loads) shall be maintained.
- (3) Management and control of the battery state-of-charge, discharge, and reconditioning functions shall be maintained.

3.2.3 Attitude, Translation, and Pointing Control (ATPC)

- (1) The ATPC function shall be capable of autonomous attitude reference acquisition and reacquisition.
- (2) The ATPC function shall be capable of autonomous fault detection, correction, and recovery of its subsystem elements (sensors, computers, actuators).
- (3) The ATPC function shall be capable of autonomous inertial and celestial sensor calibrations to compensate for changes and/or degradation of sensor parameters. Compensation activities shall be transparent to the payload user.
- (4) Autonomous translation control shall support stationkeeping to the accuracies required to meet mission requirements.

- (5) Autonomous attitude determination shall support commanding of antennas and payload instrumentation pointing to accuracies necessary to support the mission requirements.
- (6) The ATPC function shall be capable of autonomous attitude-control propellant management by changing operational mode and/or parameters.
- (7) The ATPC function shall be capable of high-level command and decision-making for activities such as initiation of turn for star reacquisition, translation control, and instrument pointing.

3.2.4 Data Processing

- (1) The spacecraft data-processing function shall be provided with adequate parametric data from spacecraft sensors and subsystems so that spacecraft performance, resource status, and integrity can be determined onboard.
- (2) The spacecraft data-processing function shall be capable of performing from available parametric data the necessary diagnostic analysis required to assess the performance, resource status, and integrity of the spacecraft.
- (3) The spacecraft data-processing function shall be capable of implementing from available parametric data the necessary diagnostic analysis required to assess the performance, resource status, and integrity of the spacecraft.
- (4) The spacecraft data-processing function shall be capable of storing a) pertinent parametric data, b) diagnostic analysis results, c) data reflecting control actions taken, and d) response data to autonomous control actions necessary to allow ground reconstruction of the spacecraft state for time intervals up to six months. These data shall be available for ground assessment upon request.

3.2.5 Payload

- (1) Failure modes within the payload function shall not propagate into other spacecraft functions.
- (2) Redundant functional command and control paths through the payload function shall not be inhibited by autonomous features.

3.2.6 Telemetry, Tracking, and Command (TT&C)

- (1) The TT&C function will allow a message to be transmitted to the ground at the first opportunity following an autonomous management activity.
- (2) The TT&C function shall be prepared to receive ground commands at any time while in the autonomous state.
- (3) The TT&C function shall be capable of transmitting, at the discretion of ground control, normal telemetry and ranging while in the autonomous state.

3.2.7 Navigation

- (1) The spacecraft shall maintain the orbit within specified limits for 60 days from the last required initialization.
- (2) If ground supervisory contact is not re-established after 60 days of autonomous navigation, the spacecraft will continue to operate within acceptable limits even if the navigation function performance is degraded. Performance degradation will be measured by the effects of degraded orbit control on payload performance.
- (3) Spacecraft orbit state or orbit-derived data shall be available to other onboard subsystems and/or user ground facilities as required. Potential examples: Sun-Earth-vehicle angle to attitude control; Sun-occultation predictions to attitude control and power; lunar-occultation prediction to attitude control; and station-acquisition data and antenna-pointing vector to ground.
- (4) The spacecraft shall have the capability to accept initialization-state data from the ground or an external source such as the Global Positioning System (GPS). It shall have a limited state reinitialization capability for some range of orbit parameters perturbed about the nominal operating orbit.
- (5) The navigation function shall be capable of adjusting performance limits based upon the availability of limited resources.
- (6) The navigation function shall be capable of executing a maneuver, if required.
- (7) The navigation function shall be capable of re-establishing the normal orbit, within acceptable limits, following an evasive maneuver, if required.

3.2.8 Propulsion

- (1) The propulsion function shall detect and isolate autonomously any failed or degraded thrusters and reconfigure the thruster complement to support mission functions.
- (2) The propulsion function shall detect and isolate autonomously any leaking propellant-supply components, including tanks.
- (3) The propulsion function shall manage autonomously any limited-life components (e.g., monopropellant thrusters) to meet life-time requirements.
- (4) The propulsion function shall be capable of estimating a-priori any impulse delivered to support navigation maneuvers and attitude-control functioning.

3.3 SPECIFICATION CONTENT

The Space Vehicle Development Specification format and content are described in SD-BG-6, "Space Vehicle Development Specification Preparation Guide", 1 April 1981. The vehicle level specification details fall into five major categories with the numbers in parentheses referring to the appropriate specification paragraph.

- (1) Operational Concepts (3.1.6)
- (2) Performance and Physical Characteristics (3.2)
- (3) Design and Construction - Computer Resources (3.3.8)
- (4) Major Component (Subsystem) Characteristics (3.7)
- (5) Quality Assurance Provisions (4.0)

The autonomy requirements should be allocated to this document as follows:

3.3.1 Operational Concepts

Scope and duration of autonomous operation requirements for the overall space vehicle should be specified for each phase of prelaunch or on-orbit operations.

3.3.2 Performance and Physical Characteristics

This section allows expansion of the description of autonomy operational concepts to an operating mode level of detail for the vehicle. The role of autonomy in survivability, reliability, and redundancy management should be specified in the appropriate subparagraphs.

3.3.3 Computer Resources

Ground simulation, onboard software development, audit trail processing, subsystem status monitoring, and anomaly analysis support requirements should be detailed.

3.3.4 Subsystem Level Requirements

The subsystem level autonomy implementation requirements selected from Section 3.2 of this Handbook should be placed in the appropriate paragraph.

3.3.5 Quality Assurance Provisions

Validation test requirements selected from the system level goals of Section 3.2 of this Handbook should be included.

Contents

Part III - Design Methodology

	<u>Page</u>
1. The Design Process	III-8
1.1 Spacecraft System Level Autonomous Design.	III-8
1.1.1 Establish a Mission-Specific Functional Baseline	III-8
1.1.2 Allocation of Autonomous Functions to Systems and Subsystems.	III-10
1.1.3 Assess Performance/Cost Impacts of Allocated Autonomy	III-14
1.1.4 Integrate with External Interface and Requirements	III-14
1.1.5 Establish Autonomous Baseline.	III-14
1.2 Critical Issues In Design.	III-18
1.2.1 Hierarchical Refinement of Requirements. .	III-18
1.2.2 Selection of Implementation Techniques . .	III-18
1.2.3 Autonomy Requirements Allocation Tradeoffs.	III-24
1.2.4 Operability of Autonomous Features	III-25
1.2.5 Autonomy for New Designs or Existing Designs.	III-25
1.3 Autonomy Implications for the System Development Process.	III-27
1.3.1 Program Management Concerns.	III-27
1.3.2 Specification and Documentation.	III-28
1.3.3 Reviews and Audits	III-29
1.3.4 Cost/Performance Analysis and Design . . .	III-30
1.3.5 Software Development	III-31
2. Spacecraft System Autonomous Design Techniques	III-33
2.1 System Level Architecture Techniques and Issues. .	III-33
2.1.1 Data Processing and Control Architecture .	III-34
2.1.1.1 Baseline System Functional Requirements	III-35
2.1.1.2 Identification of Autonomy Needs.	III-35

	<u>Page</u>
2.1.1.3 Autonomous Spacecraft System Design.	III-35
2.1.1.4 Centralized Executive Design . .	III-36
2.1.1.5 Distributed Processing Design. .	III-37
2.1.1.6 Common Memory Design	III-38
2.1.2 Reliability and Redundancy	III-38
2.1.2.1 Basic Approaches	III-39
2.1.2.2 Function Relationship Analysis .	III-39
2.1.2.3 Failure Mode Effect Analyses . .	III-39
2.1.2.4 Redundancy Provisions.	III-39
2.1.2.4.1 Functional Redundancy.	III-40
2.1.2.4.2 Cooperative Redundancy	III-40
2.1.2.4.3 Block Redundancy . . .	III-40
2.1.3 Fault Management	III-41
2.1.3.1 Fault Characteristics.	III-41
2.1.3.1.1 Operational Impact of the Fault.	III-41
2.1.3.1.2 Fault Definition Level.	III-42
2.1.3.1.3 External Interfaces. .	III-43
2.1.3.2 Fault Detection.	III-43
2.1.3.2.1 Direct Measurement . .	III-43
2.1.3.2.2 Indirect Measurement .	III-43
2.1.3.2.3 Inference of Fault Conditions	III-44
2.1.3.3 Fault Isolation.	III-44
2.1.3.3.1 Common Symptoms. . . .	III-44
2.1.3.3.2 Multiple Corrective Response	III-44
2.1.3.3.3 Transient/Permanent Fault Identification .	III-45
2.1.3.3.4 Interrelated Faults. .	III-45
2.1.3.3.5 Correction Priorities.	III-45
2.1.3.4 Fault Correction	III-46
2.1.3.4.1 Simplification of Correction Strategy. .	III-46
2.1.3.4.2 Redundant Element Availability	III-46

	<u>Page</u>
2.1.3.4.3 Block and Functional Redundancy	III-47
2.1.3.4.4 Fault Impact Across Interfaces	III-47
2.1.4 Software Fault Detection Implementation. .	III-47
2.1.4.1 Software and Faults.	III-47
2.1.4.2 Format Protocol Techniques . . .	III-49
2.1.4.2.1 Parity Checks.	III-49
2.1.4.2.2 Sparse Word Bilevel. .	III-50
2.1.4.2.3 First and Last Bits Identical.	III-50
2.1.4.2.4 Sequence of Protocols. .	III-50
2.1.4.2.5 Mode Validity Check. .	III-50
2.1.4.2.6 Heartbeat.	III-50
2.1.4.3 Performance Assessment Techniques	III-50
2.1.4.3.1 Magnitude Limit Checks	III-51
2.1.4.3.2 Sign Checks.	III-51
2.1.4.3.3 Evaluation of Constants.	III-51
2.1.4.3.4 Overflow Test.	III-51
2.1.4.3.5 Correlation of Inputs. .	III-51
2.1.4.3.6 Redundant Computation. .	III-51
2.1.4.3.7 Checksums.	III-52
2.1.4.4 Logical Fault Detection Techniques	III-52
2.1.4.4.1 Execution Timing . . .	III-52
2.1.4.4.2 Loop Counter Limits. .	III-52
2.1.4.4.3 Error Count Thresholds	III-52
2.1.4.4.4 Control Action Counts. .	III-52
2.1.4.4.5 Self-Test Algorithm. .	III-53
2.1.4.4.6 Ticket Checks.	III-53
2.1.5 Algorithm Development.	III-53
2.1.5.1 Algorithm Function	III-53
2.1.5.2 Requirements	III-54

	<u>Page</u>
2.1.5.3 Critical Fault Selection	III-54
2.1.5.4 Hardware/Software Tradeoffs. . .	III-55
2.1.5.5 Algorithm Development Process. .	III-56
2.1.6 Autonomous Control and Fault Management Algorithm.	III-58
2.1.6.1 Generalized Algorithm Forms. . .	III-58
2.1.6.2 Specific Algorithms from Planetary Designs.	III-59
2.1.6.2.1 Viking Orbiter	III-59
2.1.6.2.2 Voyager.	III-62
2.2 Autonomous Navigation.	III-66
2.2.1 Functions Supported.	III-66
2.2.1.1 Instrument Pointing.	III-66
2.2.1.2 Orbital Stationkeeping	III-66
2.2.1.3 Avoidance Maneuvers.	III-67
2.2.1.4 Communications	III-67
2.2.1.5 Attitude Control	III-67
2.2.1.6 Data Annotation.	III-67
2.2.1.7 Orbital Event Predictions. . . .	III-67
2.2.1.8 Anomaly Detection.	III-67
2.2.1.9 Relative Vehicle Control	III-67
2.2.1.10 Functional Hierarchy	III-68
2.2.2 Autonomous Navigation Subsystem Description.	III-68
2.2.2.1 Navigation Measurement Sensors.	III-71
2.2.2.2 Data Editor.	III-71
2.2.2.3 Orbit Determination.	III-71
2.2.2.4 Trajectory	III-71
2.2.2.5 Maneuver Planning.	III-71
2.2.2.6 Maneuver Commands.	III-71
2.2.2.7 Clock.	III-72
2.2.2.8 Executive.	III-72
2.2.2.9 Interfaces	III-72
2.2.3 System Level Requirements and Considerations	II-72

	<u>Page</u>
2.2.3.1 Mission and System Requirements.	III-73
2.2.3.2 Spacecraft Design Requirements .	III-73
2.2.3.3 Subsystem Characteristics. . . .	III-73
2.2.3.4 Operational Environment. . . .	III-74
2.2.3.5 Ground Systems	III-74
2.2.4 Measurement Sensors for Autonomous Navigation	III-74
2.2.4.1 Inertial Sensors	III-74
2.2.4.2 Near Body Sensors.	III-78
2.2.4.3 Clocks	III-81
2.2.5 Process Measurements (Data Conditioning)	III-85
2.2.5.1 Requirements on Data Conditioning	III-85
2.2.5.2 Identification of Bad Data . . .	III-86
2.2.5.3 Performance Monitoring	III-86
2.2.5.4 Data Compression	III-86
2.2.6 Determine Spacecraft State Vector.	III-86
2.2.6.1 Requirements on OD Function. . .	III-87
2.2.6.2 Estimation Algorithm Selection .	III-87
2.2.6.3 Formulation of Models.	III-89
2.2.7 Propagate the Ephemeris.	III-92
2.2.7.1 Orbit Prediction	III-92
2.2.7.2 Forces	III-92
2.2.7.3 Coordinate Systems	III-92
2.2.7.4 Celestial Body Ephemeris	III-93
2.2.7.5 Event Prediction	III-93
2.2.7.6 Other Considerations	III-93
2.2.8 Maneuver Planning.	III-94
2.2.8.1 Functional Description	III-94
2.2.8.2 Requirements on Maneuver Function	III-96
2.2.8.3 Maneuver Techniques.	III-98
2.2.8.3.1 Transfer	III-98
2.2.8.3.2 Rendezvous	III-99
2.2.8.3.3 Stationkeeping	III-100
2.2.8.3.4 Orbit Maintenance. . . .	III-101
2.2.9 Maneuver Command Function.	III-103

	<u>Page</u>
2.2.9.1 Functional Description	III-103
2.2.9.2 Interfaces	III-103
2.2.9.3 Detailed Description	III-103
2.2.10 Verify Navigation Performance.	III-105
2.2.11 Historical Summary of Autonomous Navigation Studies	III-106
2.3 Design Requirements for Support of Validation and Testing.	III-112
2.3.1 Validation Requirements on System Design .	III-112
2.3.2 Audit Trail Requirements	III-114
2.3.2.1 Information Recording and Storage.	III-114
2.3.2.2 Implementation Logic Requirements	III-115
3. Subsystem Level Autonomous Design Characteristics. . . .	III-117
3.1 Tracking, Telemetry, and Command (TT&C) Subsystem. .	III-117
3.1.1 Functional Description and Elements. . . .	III-117
3.1.1.1 Uplink Functional Elements	III-119
3.1.1.2 Downlink Functional Elements . . .	III-121
3.1.1.3 Tracking Functional Elements . . .	III-123
3.1.2 Autonomous Maintenance Functions	III-123
3.1.3 Autonomous Fault Management Function . . .	III-125
3.2 Power Subsystem.	III-138
3.2.1 Functional Description and Elements. . . .	III-138
3.2.1.1 Batteries.	III-138
3.2.1.2 Battery Chargers	III-138
3.2.1.3 Voltage Regulators	III-138
3.2.1.4 DC-DC Converters	III-138
3.2.1.5 DC-AC Inverters.	III-138
3.2.1.6 Load Power Switching	III-139
3.2.1.7 Memory-Keep-Alive Power Supply .	III-139
3.2.1.8 Ordnance Power Switching Unit. .	III-139
3.2.2 Autonomous Maintenance Functions	III-139
3.2.3 Autonomous Fault Maintenance	III-139
3.3 Attitude Control Subsystem	III-145
3.3.1 Functional Description and Elements. . . .	III-145

	<u>Page</u>
3.3.1.1 Reaction Wheels.	III-145
3.3.1.2 Sensors.	III-145
3.3.1.3 Thrusters.	III-145
3.3.1.4 Computer/Attitude Control Electronics.	III-145
3.3.2 Autonomous Maintenance Functions	III-145
3.3.3 Autonomous Fault Management Functions. . .	III-153
3.4 Propulsion Subsystem	III-164
3.4.1 Functional Description and Elements. . . .	III-164
3.4.1.1 Propulsion Subsystem Functions .	III-164
3.4.1.2 Subassembly Grouping	III-164
3.4.2 Autonomous Maintenance Functions	III-165
3.4.3 Autonomous Fault Management Functions. . .	III-166
3.4.3.1 Fault Sensing Techniques	III-166
3.4.3.2 Autonomous Fault Management Examples	III-170
3.5 Thermal Control Subsystem.	III-173
3.5.1 Functional Description and Elements. . . .	III-173
3.5.1.1 Passive Elements	III-173
3.5.1.2 Semi-Active Elements	III-174
3.5.1.3 Active Elements.	III-174
3.5.2 Autonomous Maintenance Functions	III-175
3.5.3 Autonomous Fault Management Functions. . .	III-175

PART III

DESIGN METHODOLOGY

The design methodology for autonomous spacecraft consists of an engineering process carried out by a design team, a set of spacecraft system level architecture and design techniques, and a set of subsystem level control requirements and characteristics.

SECTION 1

THE DESIGN PROCESS

The design engineering process is concerned with the identification of specific autonomy functional requirements and the system/subsystem level implementation trade-offs needed for a candidate design. The design process section provides some rationale for the suggested approach and discusses the implications of autonomy upon the normal spacecraft design and development process.

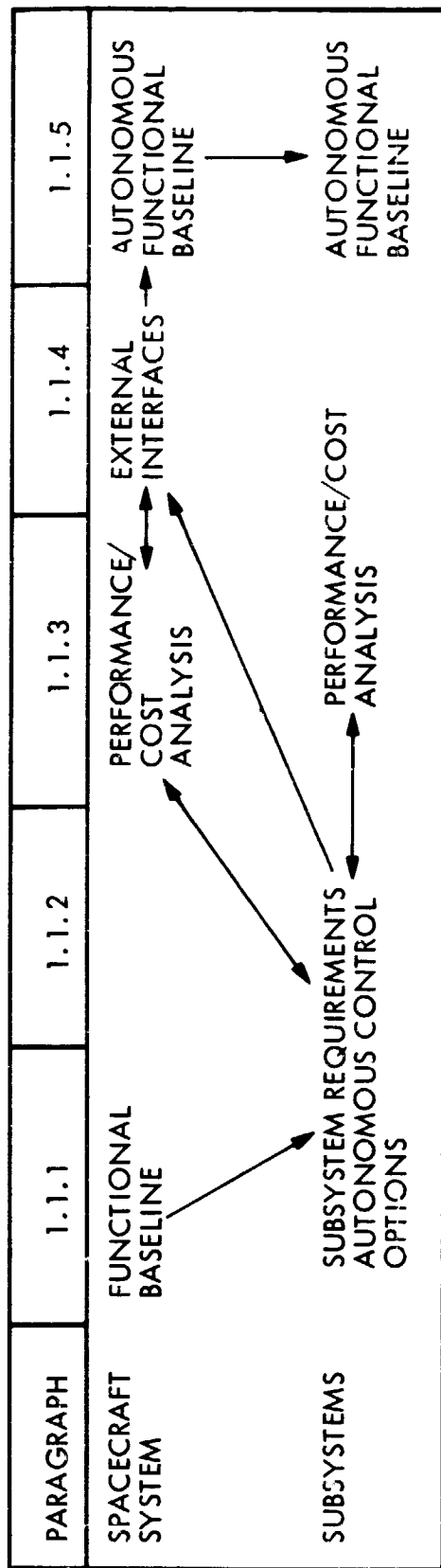
1.1 SPACECRAFT SYSTEM LEVEL AUTONOMOUS DESIGN

Spacecraft system design is not an ideal example of a process suited for top-down design methodology. The nature of the total system development process requires many implementation assumptions, of varying levels of detail, in the conceptual and mission definition phases. Consequently, the mission requirements will frequently specify or seriously constrain design features to the spacecraft system or subsystem level. This basic fact underlies the importance of specifying autonomous operations requirements at the mission level lest they be overlooked in the morass of other requirements. Mission lifetime constraints, power levels, payload data stream, orbital parameters, and a list of other requirements must be met regardless of the degree of autonomy required. A basic approach to inclusion of autonomy is to first choose the set of spacecraft functions required to perform the basic mission and then identify the functions that must be given a required level of autonomy. An iterative portion of the process concerns the resolution of system and subsystem level requirements and the selection of the appropriate level for allocation of autonomous control for each function. Table III-1 shows the system and subsystems involvement in the approach described by the following paragraphs.

1.1.1 Establish a Mission-Specific Functional Baseline

Fundamental mission requirements can be factored into a functionally oriented set of spacecraft requirements. These functional requirements on the spacecraft design can be organized in a hierarchical manner. Health and welfare and navigation can be considered to perform three high level functions: 1) Perform a set of useful services, 2) Manage resources on the spacecraft, 3) Maintain the operational integrity of the spacecraft. Each of these categories of functions will contain a series of

Table III-1. System and Subsystem Levels Involvement in Autonomous Design



NOTE: THE PERFORMANCE/COST ANALYSIS MAY FORCE RE-EVALUATION OF SUBSYSTEM REQUIREMENTS AND AUTONOMOUS CONTROL OPTIONS.

specific functions which can be specified to a level of performance required to satisfy the mission requirements. This categorization of functions should be compiled without regard to allocation to conventional subsystems or to autonomous control. The intent is to identify all functions that must be accomplished without regard to the control authority being located on the ground or spacecraft. This practice must not be constrained at this point by "known" details of subsystem implementation. The product that results from this is a complete list of functions required of the spacecraft system baseline that characterizes the design. Examples of a hierarchical breakdown of functions for a typical spacecraft mission are given by Figures III-1, III-2, and III-3.

1.1.2 Allocation of Autonomous Functions to Systems and Subsystems

This step occurs along with the development of a traditional spacecraft design architecture. Two processes are accomplished and integrated to identify autonomous design options.

A set of functional spacecraft subsystems is defined and the functional breakdown and performance requirements derived in Topic 1.2.1 are allocated to these subsystems. Each subsystem engineer then describes each function in each of the three categories by the control structure needed for implementation. Control requirements for each function are described in terms of sensing the need for action, directing actions based on analysis of the sensed data, and acting to accomplish the control action. The control structure steps of sense, direct and act may be conceptually located on the ground or the spacecraft. Figures III-4, III-5, and III-6 provide examples of detailed functional breakdowns for power, attitude control, and command/telemetry functions. The lower levels of these functions are the items that must be examined for autonomous control requirements.

The second process is to determine the options for autonomous control allocation. The autonomous operations requirements for the spacecraft are used to determine whether each step of the control structure must be autonomous. In practice, the steps are further categorized as:

- Category I - Required to be autonomous
- Category II - Not required, but increases performance, lifetime, etc.
- Category III - Autonomy not required or not possible

The integration of these processes produces a selected list of autonomous control functions which must be provided at system or subsystem levels to meet autonomy requirements. The two additional categories are those which can be provided if cost effective or responsive to other mission requirements, and those which can not be autonomous by their nature.

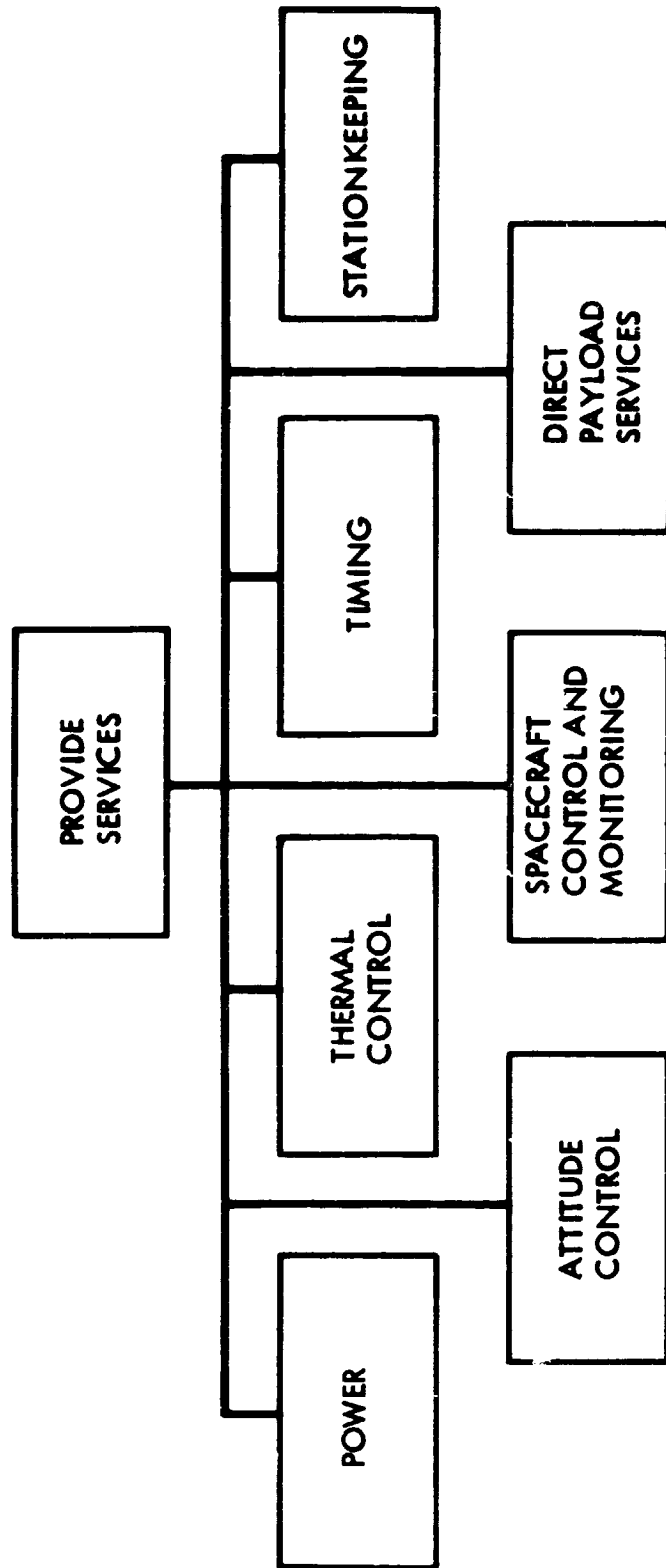
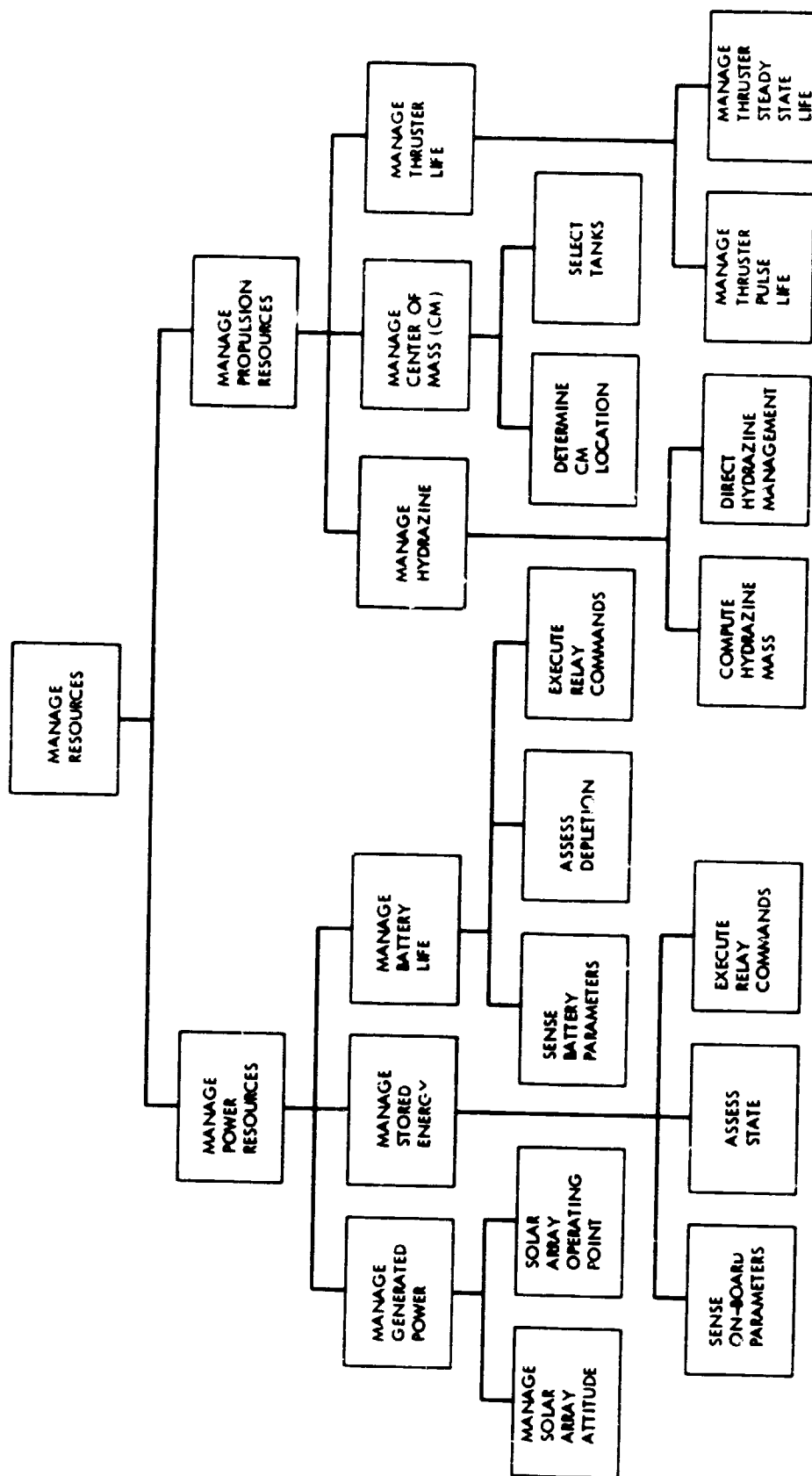


Figure III-1. Services Functional Hierarchy



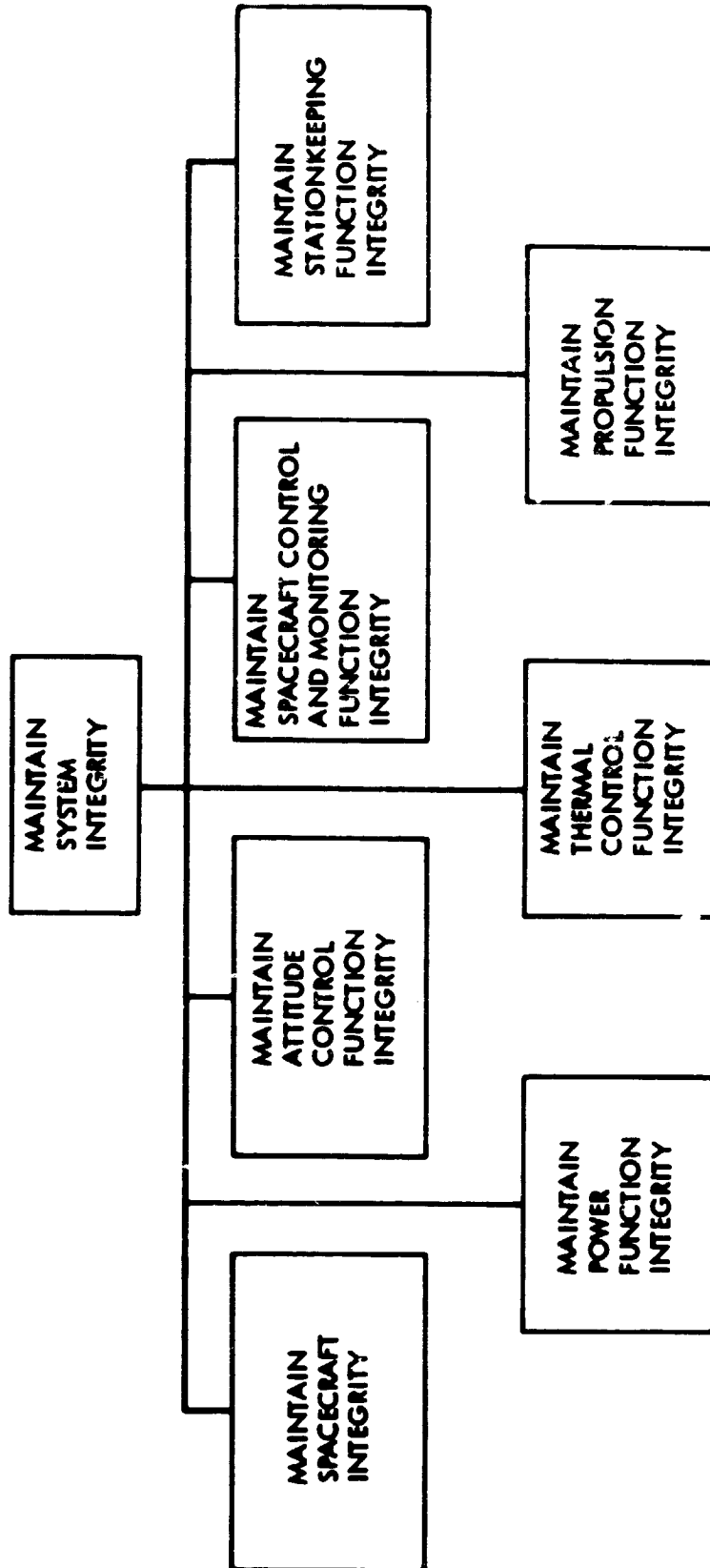


Figure III-3. Integrity Maintenance Functional Hierarchy

The result of this step is the identification of the required autonomous control structure for each spacecraft function. The required autonomous features may then be analyzed for implementation at system or subsystem levels.

1.1.3 Assess Performance/Cost Impacts of Allocated Autonomy

A specific set of autonomous functions can usually be satisfied by several allocations of control responsibility between system and subsystem resources. Topic 2.1.1 of Part III discusses the process of developing a control and data processing architecture that will support autonomous control functions. A variety of trade-offs are possible for any given architecture. The designer must perform an analysis of the cost and performance of his options and choose an approach that satisfies his mission and programmatic constraints as well as autonomy requirements. Cost impact on overall mission operations must be assessed as well as cost and schedule impacts on the spacecraft system. This may require the attention of the contracting agency to ensure that all operational costs and issues are properly considered.

1.1.4 Integrate with External Interfaces and Requirements

The spacecraft architecture and allocated autonomous control functions must be integrated with the evolved design of the ground segment, mission operations concepts, validation requirements, and engineering specialty requirements such as reliability and survivability. The result of this process may be to conclude that the chosen configuration does not meet other requirements external to the spacecraft functions. Such a conflict may be resolved by a new allocation of autonomous control functions, changes to the spacecraft requirements, or some combination of these.

1.1.5 Establish Autonomous Baseline

The selected design architecture and autonomous control structure must be documented as the baseline configuration. The allocation of autonomous functions in the design should be traceable to mission or spacecraft requirements. Performance specifications for the autonomous functions should be developed in sufficient detail to aid hardware/software implementation tradeoffs, and required system, subsystem, and external interfaces must be defined.

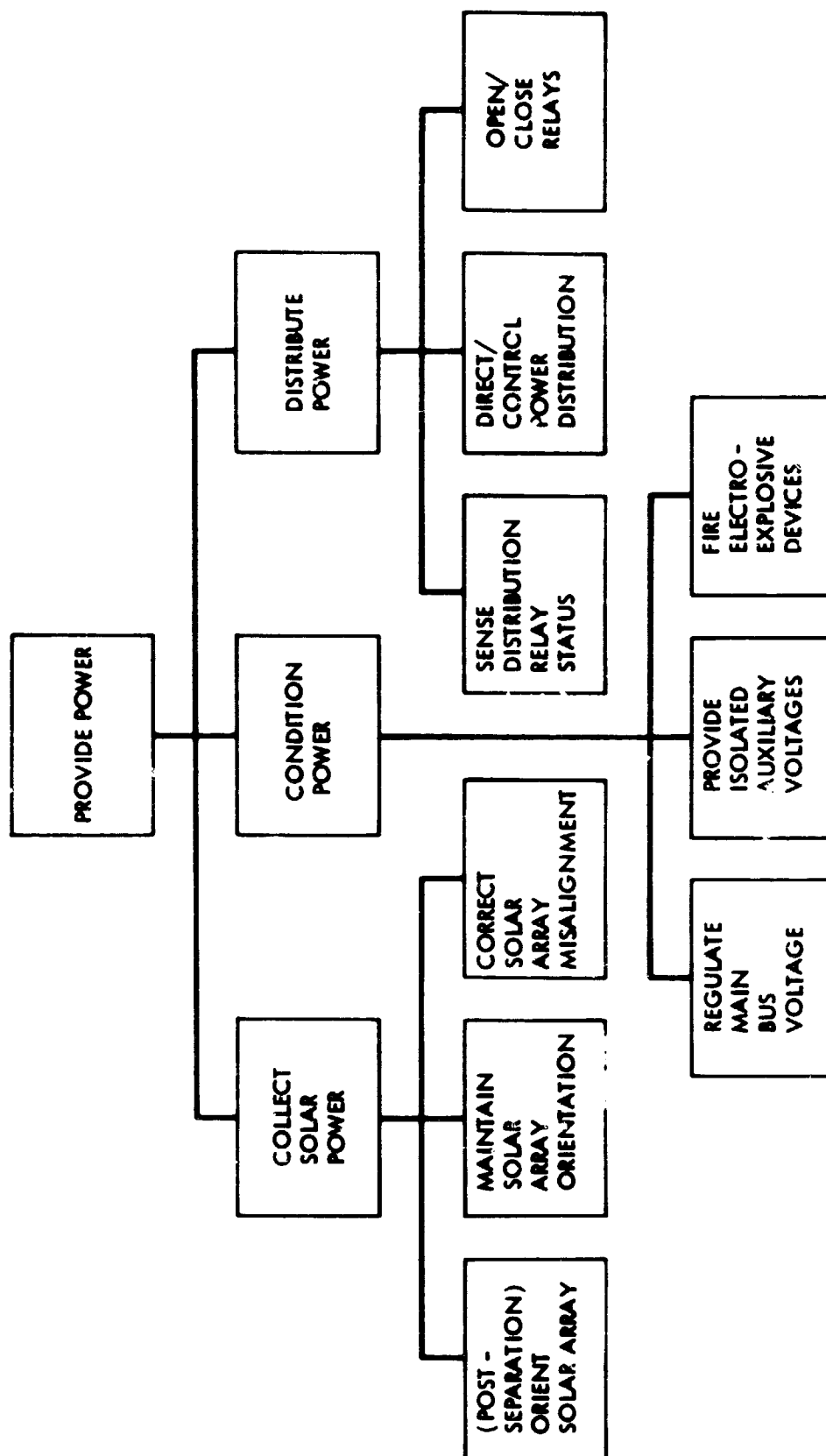


Figure III-4. Power Services Functional Hierarchy

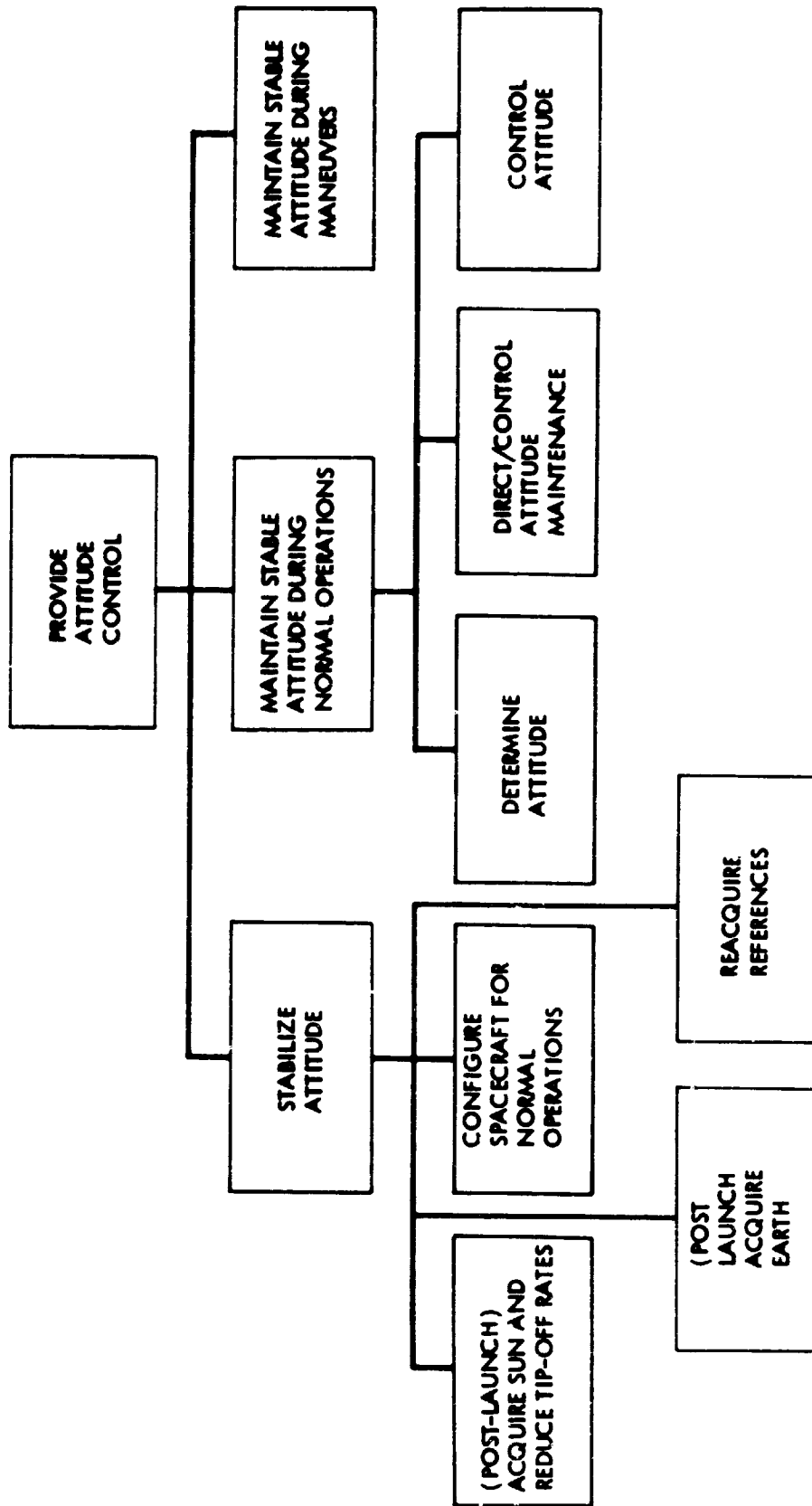


Figure III-5. Attitude Control Service Functional Hierarchy

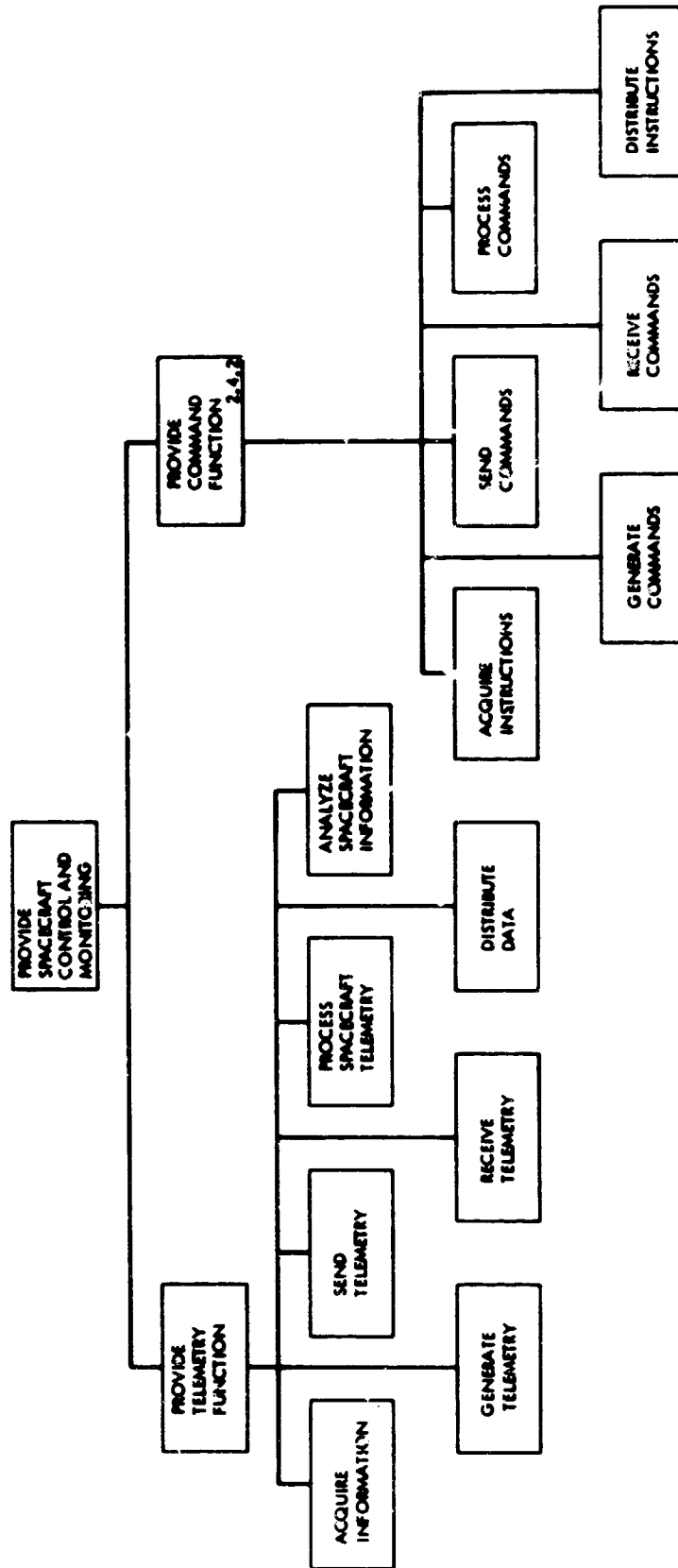


Figure III-6. Spacecraft Control and Monitoring Service Functional Hierarchy

1.2

CRITICAL ISSUES IN DESIGN

The design process specified in the topics of Section 1.1 is based on the fundamental need to define functions required for mission success before attempting the allocation of autonomy. The success of this implementation process is dependent upon several important details. This section discusses some of these details and suggests how the context of individual programs may modify them.

1.2.1

Hierarchical Refinement of Requirements

Section 1.1 raised the point that spacecraft system design is not ideal for a top-down design approach. Some form of top-down approach, however, greatly simplifies the analysis of functional requirements to define options for implementing autonomy. The method chosen has proven valuable in categorizing the necessary mission functions, first by their application in the mission (services, resource management, and integrity maintenance), and second by the need for their provision as autonomous functions. The three functional categories are not intended to fit into a traditional spacecraft functional architecture, but rather to emphasize the nature of the function's criticality to autonomous operations requirements. The further breakdown into Categories I, II, and III are intended to further clarify the design options by separating out those functions which can provide lifetime or other second level benefits (II) from those clearly needed to achieve autonomous mission requirements (I) and those which are not required to be autonomous (III). The Category II functions can be considered for implementation if their benefits clearly exceed costs, other mission level requirements are satisfied, or the implementation is easy using resources which will be provided for the Category I functions. A further breakdown of Category I and II functions by control structure (sense, direct, act) requirements will highlight the need for on-board sensing, logic, and control requirements for each function. This will aid in the assessment of the cost (money, weight, power, etc.) of implementing a specific allocation of autonomous functions. Table III-2 provides an example of the result of applying this technique to a limited subset of functions required of the DSCS III spacecraft. This approach is equally applicable to new or existing designs. In one case, the functions are derived in the process, in the other they are characteristics of the given design.

1.2.2

Selection of Implementation Techniques

The selection of implementation techniques involves tradeoffs between system or subsystem control authority and between hardware or software implementation. The next topic deals with many of the issues regarding allocation of control authority. Its primary effect upon technique selection is on the number of interfaces required to perform a function and the scope of control that is provided to a function. System executive control tends to deal with problems that have a large scope of control, impact different subsystems, or are spacecraft critical. Many techniques applicable to system executive control are also applicable to control at the subsystem level.

Table III-2. List of Current DSCS III Functions by Autonomy Level and Category:
(a) Services, (b) Resources, (c) Integrity (Sheet 1 of 4)

LEVEL	(a) SERVICES		
	CATEGORY		
	I	II	III
0	<p>PROCESS LOCATION MEASUREMENTS (NAV) DETERMINE S/C LOCATION (NAV) PROPAGATE EPHEMERIS (NAV) PLAN MANEUVERS (NAV) GENERATE MANEUVER COMMANDS (NAV) SELECT TANKS (NAV) SELECT THRUSTERS (NAV) CONTROL CATALYST BED HEATERS (TCS)</p>	<p>SPACE SEGMENT TRACKING (NAV) VERIFY NAVIGATION PERFORMANCE (NAV)</p>	
1			<p>REORIENT GDA (ACS) RECONFIGURE MBA (ACS)</p>
2	<p>SENSE DISTRIBUTION RELAY STATUS (EPDS) DIRECT/CONTROL POWER DISTRIBUTION (EPDS) OPEN/CLOSE RELAYS (EPDS) DIRECT/CONTROL MANEUVERS (ACS)</p>		
3	<p>ACQUIRE EARTH (ACS) CONFIGURE S/C (ACS) REACQUIRE REFERENCES (ACS) DETERMINE ATTITUDE (ACS) ENABLE SURVIVAL HEATERS (TCS) ENABLE CONTROL HEATERS (TCS) ENABLE SURVIVAL THERMOSTATS (TCS) ENABLE OTW? HEATERS (TCS) CONTROL ELECTRIC COMP. HEATERS (TCS) CONTROL SURVIVAL HEATERS (TCS) ACQUIRE INFORMATION (IT & C) GENERATE TELEMETRY (IT & C) PROCESS COMMANDS (IT & C) DISTRIBUTE INSTRUCTIONS (IT & C) WARM UP CATALYST BED (ACS) FIRE THRUSTERS (ACS) MAINTAIN ATTITUDE IN MANEUVERS (ACS)</p>	<p>SEND TELEMETRY (IT & C) RECEIVE COMMANDS (IT & C)</p>	

Table III-2. List of Current DSCS III Functions by Autonomy Level and Category:
(a) Services, (b) Resources, (c) Intergity (Sheet 2 of 4)

LEVEL	CATEGORY		
	I	II	III
4	<p>ORIENT SOLAR ARRAY (ACS)</p> <p>MAINTAIN SA ORIENTATION (ACS)</p> <p>ACQUIRE SUN (ACS)</p> <p>DIRECT/CONTROL ATTITUDE (ACS)</p> <p>CONTROL ATTITUDE (ACS)</p> <p>PROVIDE TIMING (ACS)</p>		
5	<p>REGULATE MAIN BUS VOLTAGE (EPDS)</p> <p>PROVIDE AUXILIARY VOLTAGES (EPDS)</p> <p>FIRE ELECTRO-EXPLOSIVE DEVICES (EPDS)</p>		
(b) RESOURCES			
0	<p>ASSESS POWER STATE (EPDS)</p> <p>ASSESS BATTERY DEPLETION (EPDS)</p>		<p>SOLAR ARRAY OPERATING POINT (EPDS)</p> <p>REDUCE N/S STATIONKEEPING (NAV)</p>
1	<p>SENSE ON-BOARD PARAMETERS (EPDS)</p> <p>SENSE BATTERY PARAMETERS (EPDS)</p> <p>DETERMINE C OF M LOCATION (PROP)</p>		
2	<p>EXECUTE RELAY COMMANDS (EPDS)</p> <p>EXECUTE RELAY COMMANDS (EPDS)</p> <p>SELECT TANKS (ACS/PROP)</p>	<p>COMPUTE HYDRAZINE MASS (PROP)</p> <p>DIRECT HYDRAZINE MNGT. (PROP)</p> <p>MANAGE THRUSTER PULSE LIFE (PROP)</p>	<p>MANAGE THRUSTER STEADY-STATE LIFE (PROP)</p>
3			
4			
5	<p>MANAGE SOLAR ARRAY ATTITUDE (ACS)</p>		

Table III-2. List of current DSCS III Functions by Autonomy Level and Category:
(a) Services, (b) Resources, (c) Integrity (Sheet 3 of 4)

(c) INTEGRITY			
LEVEL	CATEGORY		
	I	II	III
0			
1	PROTECT POWER BUS (EPDS)		
2	BATTERY CHAIN FAILURE (EPDS) SECONDARY CONVERTER FAILURE (EPDS) SINGLE LOAD SWITCH FAILURES (EPDS) FAILED BATTERY CHAIN (EPDS/ACS) SAD POT FAILURE (ACS) SENSE DEVICE STATE (ACS) ATTITUDE DURING ECLIPSE (ACS) YAW RATE REDUCTION (ACS) REACTION WHEEL FAILURE (ACS) ACE FAILURE (ACS) EARTH SENSOR FAILURE (ACS) SUN SENSOR FAILURE TELEMETRY CONTINGENCY MODES (ACS) MAINTAIN THERMAL CONTROL (TCS) MAINTAIN INFO ACQUISITION (TT & C) MAINTAIN TELEMETRY GENERATION (TT & C) MAINTAIN COMMAND PROCESSING (TT & C) MAINTAIN COMMAND DISTRIBUTION (TT & C) MAINTAIN THRUSTER HEALTH (PROP) MAINTAIN PROPELLANT SYSTEM (PROP)	BASELINE ENERGY STORAGE (EPDS) ALL AXES SUN CONTROL (ACS) MAINTAIN TELEMETRY TRANSMISSION (TT & C) MAINTAIN S/C COMMAND RECEPTION (TT & C) MAINTAIN TRACKING FUNCTION (TT & C)	RAM PATCH (ACS)
3	ISOLATE LOAD FAULTS (EPDS) SA OR SHUNT DISSIPATOR FAILURE (EPDS) BATTERY OPERATIONS INTEGRITY (EPDS) LOSS OF EARTH PRESENCE (ACS)		GROUND OVERRIDE (ACS)
4	DATA TRANSFER HANDSHAKES (ACS) PROTECT FROM FALSE COMMANDS (ACS) CHECK PARAMETER STATES (ACS)		

Table III-2. List of Current DSCS III Functions by Autonomy Level and Category:
(a) Services, (b) Resources, (c) Integrity (Sheet 4 of 4)

LEVEL	CATEGORY		
	I	II	III
5	REG ELECTRONICS/QUAD RELAY FAILURE (EPDS) NUCLEAR EVENT (ACS) PROTECT EARTH SENSOR (ACS)		

LEGEND:

ACS = ATTITUDE CONTROL SUBSYSTEM
 EED = ELECTRO-EXPLOSIVE DEVICE
 EPDS = ELECTRIC POWER DISTRIBUTION SUBSYSTEM
 GDA = GIMBAL DISH ANTENNA
 MBA = MULTIBEAM ANTENNA
 NAV = NAVIGATION
 POT = POTENTIOMETER
 PROP = PROPULSION SUBSYSTEM
 RAM = RANDOM ACCESS MEMORY
 SAD = SOLAR ARRAY DEVICE
 TCS = THERMAL CONTROL SUBSYSTEM
 TLM = TELEMETRY
 TT & C = TRACKING, TELEMETRY, AND COMMAND SUBSYSTEM

Software - hardware implementation is an option for each of the sense, direct, and act steps of the control structure. Hardware will always support the sense function to some degree. The major support is to hardware a specific sensing function to trigger the direct/act steps. The maximum software involvement would probably be in the extraction of values from an engineering telemetry or payload data stream. Hardware sensing provides the most direct approach to sensing a specific event. Its reliability is dependent only on the hardware components and the reasonableness of the sensed quantity as an indicator of the desired condition. The primary disadvantage is inflexibility. Software extraction for telemetry data allows access to a wide range of measurements without hardware instrumentation penalties. Reprogrammability of the extraction allows for adaptability to cover changes in the quantities to be sensed after the hardware is complete. Implementation in a control structure requires an interface between the telemetry function and a control authority.

Logic processes should definitely be implemented in software unless there is no possibility of their change at all. Trigger levels for control actions and fault management routines should be modifiable to account for a wide variety of conditions which can only be appreciated with flight experience. Higher than nominal noise levels, subtle system level interactions between subsystems, hardware faults, and aging of components all lead to conditions that may require a change in trigger levels for sensed data.

The direction step of the control structure is possibly the best candidate for implementation by software logic. This allows flexibility to change the conditions of response to a sensed action while utilizing the same or new sensed data. An alternative to the direction logic is use of an inferential logic that assumes a specific condition exists upon receipt of sensor notification. Thus the sense step directly invokes a specific action response. This is a compact approach, very easy to implement in software, but modifiable if experience shows the need for additional logical processing upon receipt of the sensed data. There are classes of faults or control actions that are not suited to this inferential direction due to their complexity. The example algorithms for celestial reference reacquisition in Appendices B and C are good examples of this type of circumstance. The process is simplified by using priority logic to decide on the order of execution of three separate responses. The more complex sensing of additional data relevant to each control problem is left to each of the routines. This allows the most serious issue to be dealt with immediately and the details of directing its control process to be separated from those of the less serious possibilities.

The action step of the control structure consists of providing the responses to control commands selected by the direct step. The actual control action itself will normally be a hardware implementation. A complex control action, however, may involve preconditioning spacecraft subsystems to accept changes in power distribution or thermal conditions. Some devices may be turned off to increase the power margin available for transients, while heat producing devices may be configured to different operating modes. Such complex control actions should be commanded through programmable sequences. This allows the sequences to be modified in flight to incorporate lessons learned from their use, or changes in the spacecraft operating conditions caused by failures or end-of-life performance.

1.2.3

Autonomy Requirements Allocation Tradeoffs

Allocation of autonomous functions between system and subsystem resources is dependent upon the level of autonomy to be provided and the relative role of system executive control and subsystem control for the autonomous features. The hierarchical breakdown of autonomous control structure advocated in Section 1.1 aids in the mechanics of the process and makes it easier to examine the nature of required autonomous functions. Some functions will be sufficiently unique to a subsystem that it only makes sense to provide for their control as a subsystem responsibility. Other functions, particularly those involving maintaining the integrity of spacecraft functions (i.e., fault management), will have such a broad scope of control that they must be located in some sort of overall executive control authority. There are, however, many options between these two.

Topic 2.1.1 in Part III addresses the subject of control and data processing architecture in detail and should be used as a guide in the selection of a system architecture. Subsystem designers, however, need a system level definition of the policies for system/subsystem control allocation. Projected executive control responsibilities, resources, and interface protocols should be determined to aid subsystem designers in the development of their design requirements. Allocation of the sense, direct, act functions of the control structure is one prospect for a system level design policy. An example is the provision of a central programmable executive control authority. Subsystems may be responsible for sensing fault conditions, notifying the central executive controller of faults, and providing telemetry data for implementing normal autonomous control functions. The central executive controller provides all direction and action implementation through software logic and stored command sequences. The Viking Orbiter, described in Appendix A, had this design implementation.

The types of faults to be protected against and the degree of protection to be provided are other topics for system level policy definition. Careful selection of system responsibilities for control and fault management can provide a high degree of protection for less effort than required if responsibilities are delegated completely to the subsystem level.

Power, mass, risk, and cost constraints allocated to a subsystem will influence the selection of implementation techniques. These allocations should be made on the basis of the previously defined system architecture and rules for autonomy allocation between the system and its subsystems. Autonomy requirements add to power, mass, and cost at the system level. The effect of individual subsystems will vary with the autonomous control incorporated within the subsystem. Impacts of adding control and fault sensing capability to a subsystem are not as large as providing processing capability to implement direct and act functions as well. A system level design policy is needed to define the scope of the subsystem designer's responsibility.

1.2.4 Operability of Autonomous Features

Ensuring that the implementation of autonomy results in an operable spacecraft that meets or exceeds autonomous goals and requirements is a major systems engineering responsibility. The hierarchical approach to identifying autonomous control requirements and system level design policies for implementation is intended to make this process easier. It is still necessary to review subsystem interface designs for both direct and subtle impacts on the autonomy of the overall system. Subsystem level design decisions that detract from autonomy and increase reliance on ground support are often not visible at higher levels of responsibility until late in the process. Critical Design Reviews (CDR's) can reveal such problems, though they may not surface until system integration and validation testing. This is sufficiently late in the design process that serious cost and schedule impacts may result from resolving the problems. Care must be taken to examine design decisions such as sensor visibility restrictions and subsystem operating mode characteristics early in the design for their impacts on other subsystems and overall autonomous operability. A decision that simplifies design and construction of one subsystem may well have adverse effects on autonomous functions of others. Attitude control, propulsion, and navigation are particularly susceptible to this sort of interdependence.

Ground related requirements including the earth based test environment must receive particular attention. The development of ground support requirements may suffer from lack of early visibility into spacecraft design details and operating characteristics. The difficulty is compounded if the ground operations agency and its contractors lack flight experience with complex autonomous spacecraft. The process is aided by providing the program policies and goals for autonomy to those responsible for the ground segment. A set of spacecraft requirements on the ground segment should be provided as early as possible, preferably with the spacecraft system requirements package. This should provide a set of autonomy related requirements that are needed for ground control and serve as a firm requirement on the space/ground interface. Command sequence generation, payload data processing, and spacecraft simulation are traditional areas for requirements. Software development support for on-board routines, downlink processing and analysis support for memory dumps and audit trail, and simulation requirements for programmable on-board subsystems are items that arise with the addition of autonomy. The ground segment requirements must consider impacts on both facilities and operating philosophy. Those responsible for the spacecraft design and validation must consider the effects of their design on this process and realize that they will have the first opportunity to view the effects of their design on operability. With separate contractors for space and ground segments, a major part of this systems engineering responsibility will fall upon the contracting agency or agencies. In such situations, penetration into the details of the design process must consider autonomous operation impacts as well as traditional interface details.

1.2.5 Autonomy for New Designs or Existing Designs

Two major factors influence the difference in providing autonomy to an existing design versus integrating it with a new design. The first of

these is the role of control architecture and hardware inheritance. The second is the role of mass, power, and cost constraints.

Autonomous control authority implies data processing and stored command sequence capability that may not exist on a non-autonomous spacecraft or may not provide for a simple implementation of autonomous control structures. The DSCS III design assessment of document SD-TR-81-87 provides a direct example of the application of autonomous design methodology to an existing system. Allocation of autonomous functions from existing support functions produced a series of design options that must be examined for trade-offs between benefits and constraints. Selection of an implementation option is followed by the process of designing the new autonomous control features and integrating them with the existing design. A new design allows the advantage of designing the control architecture and autonomy implementation as an integral part of the spacecraft, providing implementation options and features that might not be available otherwise.

Inherited mass, power, and processing features of an existing design tend to constrain the addition of autonomous control. Simple software routines for control algorithms offer the cheapest approach to autonomy for an existing design. This application, however, is limited by existing architecture, memory margins, and control authority allocation. Replacing an existing computer or upgrading memory to provide more space for software have severe design and validation impacts beyond direct mass, power, and cost of new hardware. Examples of some of the direct impacts are given in Volume I of SD-TR-81-87 as assessed for the current DSCS III design. The proposed autonomy options are described in more detail in Volume III.

The same types of constraints will arise in integrating autonomy into a new design. The designer has the freedom, however, to utilize autonomous control resources in the accomplishment of other mission objectives. Additionally, the positive impact of autonomy on reliability may ease constraints on redundancy of some equipment or implementation of safing features. Design of a common digital data bus for the spacecraft and integration of system and subsystem control requirements offer the designer challenging options for providing autonomous control while improving on mass and power over traditional design techniques.

A point of particular interest is the use of a Redundancy Management Subsystem (RMS) to add autonomous control to an existing design. The concept is evaluated in detail in Section 4, Volume III of SD-TR-81-87. In summary, the RMS consists of a fault tolerant processor added to a spacecraft as a separate subsystem. It utilizes software algorithms to access the conventionally designed telemetry stream, select engineering data under software control, and analyze the data to determine the need to issue pre-stored command sequences to the standard command subsystem. The intent of the design described was to provide integrity maintenance through management of redundant spacecraft resources. A bit of thought and consideration of the Viking Orbiter Computer Command Subsystem (CCS) example in Appendix A shows that this concept could easily be sized to handle a wider variety of autonomous control functions than redundancy management.

1.3

AUTONOMY IMPLICATIONS FOR THE SYSTEM DEVELOPMENT PROCESS

The current process for system development has evolved over years of experience to provide for orderly implementation of mission requirements while providing specialized attributes such as reliability and survivability. Autonomy is also a system attribute whose requirements must be integrated with mission requirements in a total system design. The scope of its impact on design, validation, and flight operations makes it necessary to specify it uniquely and track the progress of its implementation in the system engineering process. The techniques of implementation have some profound impacts on spacecraft system architecture, software requirements, and flight operation support. These details must be identified and managed to provide the product necessary for an effective operational system.

1.3.1

Program Management Concerns

Management of a program involving autonomous spacecraft differs in detail rather than process from one involving non-autonomous spacecraft. The mission conceptual and mission design phases of the program must consider the requirement of autonomy as an attribute on the same level as reliability and survivability. Project level policies and goals must be established in the context of the total mission to guide contractors and designers of space and ground segments. The acquisition process requires careful specification and management of implementation details to insure that the spacecraft design meets autonomy requirements, the ground segment provides supporting functions, and the operations plan properly utilizes autonomous operating functions.

The Program or Project Office should consider the effects of autonomy upon contractual relationships as well as upon technical implementation. Cost and risk aspects of autonomy as seen by a contractor include:

- (1) Autonomy is a new attribute whose implementation risks are not well understood.
- (2) Life cycle cost benefits of reducing ground support do not have a direct beneficial effect upon the space segment acquisition phase of the life cycle.
- (3) Increased front end costs to the space segment will result from initial implementation of autonomy.
- (4) Cost reductions to a contract involving only the space segment may be made at the sacrifice of autonomous functions or by choosing less flexible implementations.

Several measures may be taken to lessen the likelihood of programmatic difficulties from these factors.

- (1) Provide complete and detailed specification of autonomy requirements that are expected to benefit life cycle costs or provide high priority operational benefits. Specify implementation requirements in addition to autonomy scope and duration.
- (2) Provide ground test, on-orbit and lifetime performance incentives directly related to measurable performance of autonomous functions.
- (3) Give autonomy a sufficiently high priority in program requirements that there is no tendency to defer autonomous operating features to ground operations as a means of reducing costs to the flight segment contractor.
- (4) Consider the effects of risk upon the type of contract employed. A new spacecraft designed for autonomy from the start may be seen as having more risk than addition of autonomy features to an existing spacecraft design.

Software development and maintenance and increased data processing in the spacecraft design offer an additional managerial challenge. Spacecraft computers and/or microprocessors offer the most flexible and reliable means of implementation of autonomous control functions for a complex spacecraft. Hardware expertise must be provided to handle the provision for programmable logic and digital control interfaces. The software development effort associated with an autonomous spacecraft will be higher than for non-autonomous designs and will involve the spacecraft directly rather than being a ground based implementation. Configuration management of the on-board software, division of algorithm design and programming responsibilities, and continuing support in flight are all concerns that arise with the increased importance of the onboard software.

The following topics address these and other programmatic issues in more detail.

1.3.2 Specification and Documentation

Autonomy impacts program documentation in both content and magnitude. Autonomy requirements and their implementation must be specified, reviewed, and configuration managed as any other requirements. The inclusion of additional on-board software and supporting ground facility capability will result in more software peculiar documents than exist for current spacecraft designs. Processor hardware documentation will be substituted for or added to traditional hardware documents.

Autonomy requirements should be documented as top level policies and system requirements as detailed in Part II of this Handbook. A Request for Proposal at any stage of mission concept, mission definition, or acquisition should be used as a vehicle for expressing the contracting agency's policies on autonomy to the contractors. The same policies, plus supporting materials on design goals should be furnished to contractors or agencies involved with acquisition and operation of a ground segment.

The material of Part II of the Handbook discusses the specification of high level autonomy requirements for the Space System and Space Vehicle. The implementation of these autonomy requirements must be documented by the contractor in Functional Requirements to the subsystem level. This process is not unique to autonomy, but the autonomous functions, their hardware and software implementation, and their effects on the spacecraft operation must be clearly documented and traceable as autonomous functions.

1.3.3 Reviews and Audits

Preliminary and Critical Design Reviews (PDR, CDR) offer invaluable opportunities to view the progress of autonomous design features. Autonomy should be a specific agenda item for these Reviews. The PDR should contain an overview of the spacecraft architectural features that support autonomous control structures, explicit statements of system level design rules for autonomy and the allocation of specific autonomous features to system and subsystem levels. Hardware/software implementation plans, functions to be protected by fault management, and the degree of computer memory and performance margin supplied should be explicitly addressed. Preliminary plans for autonomous operation in each phase of the flight mission should be addressed, with appropriate spacecraft requirements on ground facilities, procedures, and personnel included. The overall thrust of the PDR should show explicitly how autonomy requirements will be met, just as any other mission requirement. A careful review of autonomous design features is necessary at this stage of design, as it will be more costly to add new features as the design progresses.

The CDR should address the autonomous design features at a subsystem implementation level of detail. Allocation of control structure responsibilities between system and subsystem resources, system and subsystem interfaces, control algorithms for each feature, software implementation plans, validation provisions, and status of resource margins are all topics to be considered in detail. Details of subsystem design must be carefully examined for unexpected impacts on the autonomous operation of other subsystems and the spacecraft system as a whole.

The Functional Configuration Audit will review the results of validation testing to insure that all autonomous operation requirements are met. Most requirements may be tested at the system or subsystem level. Some requirements may not be testable except in flight. These should be validated by simulation and analysis, supported by results of subsystem functional tests. It is rarely possible to test all logical paths of a complex autonomous function. The validation process will have identified critical functions and the principal operating modes of the spacecraft and these will provide a basic set for auditing the achievements of the design. The Functional Configuration Audit should insure that features critical for the use of ground operations have been properly validated as well as those critical to spacecraft operation and safety.

The Physical Configuration Audit for an autonomous spacecraft design should not differ from that required of any other spacecraft.

1.3.4 Cost/Performance Analysis and Design

The cost of implementing autonomous functions can be measured in terms of weight, power, and complexity as well as dollars. In fact, increases in these three factors usually lead to cost impacts on the spacecraft system to support them. Cost and performance analysis is important to both program planning and implementation. As yet there is no well-defined set of cost data points to relate implementation costs to the level of autonomy achieved or to reflect the effects of autonomous design and operation on life cycle cost. Consequently, the initial effort must fall upon the direct implementation costs of specific features in the spacecraft design. Life cycle costs are highly dependent upon ground operations costs, and these tend to be ill-defined or supplied at a specific level of effort. They shall have to be attacked as a separate issue once an appropriate level of spacecraft autonomy is operationally available. The most direct approach to their reduction is through mission level design and operations requirements that deliberately specify autonomy leading to decreased support requirements.

Design costs and performance benefits become visible at the spacecraft system level when the allocation of autonomous functions is performed (Topic 1.2.2). System design rules and policies should call for specific system/subsystem level techniques that provide the most effective scope of autonomous control with the least impact. This process is exemplified by the selection of critical functions to be protected at the system level without regard to the location of a failure. The Command Loss and Radio Frequency Loss algorithms from the Voyager program provide examples of control implementation in software with minimum impact on spacecraft design. The algorithms use inferential logic or simple hardware devices for sensing, and adopt a direct/act strategy of tree switching through a software specifiable set of redundant elements until the problem is solved. This provides functional protection against a number of different failure modes with minimum impact on spacecraft system resources or subsystem designs. Allocation of sensing, direction, and action steps of the control structure can also impact costs. Sensing of faults or operating characteristics may be easy at the subsystem level, but direction and action may be more easily implemented by a control executive.

These system and subsystem implementation details should be examined as cost trade-offs in terms of weight, power, complexity, and reliability. A set of options should be constructed to provide the required level of performance for the autonomous baseline design. The choice of the desired option should consider the spacecraft design impacts and potential impacts on ground operations in terms of ease of operation of the spacecraft and changes to the level of support required. The DSCS III Assessment Volumes I and III provide a high level overview and detailed example of the type of design options (SD-TR-81-89) to be considered.

1.3.5 Software Development

Software to support autonomous spacecraft will be required on-board, and on the ground. The autonomous control structure is best implemented on the spacecraft through programmable logic that can be specified through data base tables. This provides for maximum flexibility without recompilation of source code and its associated revalidation requirements. Ground software may support system/subsystem level simulation, audit trail processing, on-board software development and configuration management, and performance analysis.

Ground support software requirements for support of autonomous spacecraft will differ from traditional ground software in function, but the development process will not be appreciably affected. New or expanded functional requirements will arise for telemetry processing, health status analysis, spacecraft simulation, and software development for on-board computers.

Support of the uplink process may require simulation software to allow the assessment of the validity of programmed sequences and changes to on-board software before they are sent to the spacecraft. Cost and design trade-offs will be needed to determine the level of the simulation and the degree of modeling to be employed. The Viking Orbiter was represented by a full functional software simulation that evaluated the effects of all proposed commands upon the spacecraft. The increased complexity of the Voyager spacecraft in both payload and support subsystems made this approach unacceptably expensive. Simulations were limited to specific programmable subsystems without an attempt to exhaustively model the spacecraft behavior.

Uplink support will also require software development facilities to support design, coding, and validation of on-board software. Documentation, configuration management, and compiler/assembler support will have to be provided for the on-board computers or processors.

Downlink support will require the ability to process and analyze audit trail data and memory readouts as well as conventional engineering telemetry streams. Engineering telemetry processing can be performed in a conventional manner, though it might prove useful to have an analysis function that would correlate telemetry from a real-time stream with stored telemetry and audit trail recording of autonomous control actions.

Spacecraft software will require programmatic support for development and maintenance. Software development plans, policies, and standards must be selected for implementation of on-board software. Exceptions to the policies

should be devised. These pre-design activities are vital to an orderly development process. They set the requirements for the steps of the development and maintenance process and are needed for an understanding of the process by both management and implementers.

Software requirements must be developed and documented at system and subsystem levels. These may be integrated with hardware requirements in an overall system or subsystem level requirements document, but they will provide all information on functions and interfaces needed to proceed with software design. Software designs must be reviewed for compliance with algorithm requirements and compatibility with hardware design and operating constraints. Ease of incorporating changes should be a consideration in software design, with table driven logic used to implement all commanding actions.

Configuration management of delivered software will be important in internal software deliveries and in support of changes during validation test and flight support. Test results on delivered configurations should be carefully maintained to support future modification efforts or anomaly investigations.

SECTION 2

SPACECRAFT SYSTEM AUTONOMOUS DESIGN TECHNIQUES

Spacecraft system level concerns for autonomy are centered around the overall systems data processing and control architecture, the techniques available for implementing autonomous maintenance and fault management functions, the potential design of a navigation subsystem, and the requirement to design autonomous functions in a manner that allows them to be tested to validate proper operation.

2.1 SYSTEM LEVEL ARCHITECTURE TECHNIQUES AND ISSUES

Major system level issues for autonomy are:

- (1) Data Processing and Control Architecture - Providing or adding the processing capability needed for control of system and subsystem autonomous functions.
- (2) Reliability and Redundancy - Redundant functional capability, included in design for reliability, forms the basic tool which autonomous fault management controls. Redundancy design options influence the scope and nature of autonomous fault management functions.
- (3) Fault Management - Control of redundant resources is characterized by the need to detect, isolate, and repair faults over a wide range of spacecraft operating modes and in the presence of erroneous indications of faults.
- (4) Software Implementation Techniques - Software or firmware implementation of control logic is the most flexible means of providing autonomous control. A selection of potential means for detecting and isolating fault conditions is presented.
- (5) Control and Fault Management Algorithms - The logical process for implementing an autonomous maintenance or fault management function is an algorithm. A series of prospective and actual flight project algorithms are described.
- (6) Algorithm Design - The process of developing algorithms for autonomous functions is characterized from the viewpoint of design and flight experience.

2.1.1* Data Processing and Control Architecture

Computer size, weight, and power limitations restricted early design architectures to a single centralized processing capability usually associated with command sequencing. Late Mariner designs and the Viking Orbiter described in Appendix A typify this architecture. Increased subsystem complexity and the availability of microprocessors are currently driving decentralization of control and data processing capability to the subsystem and component levels. A broad spectrum of choice is possible between complete centralization of resources and complete decentralization. The Voyager design, described in Appendix A, provides a central executive control authority in the CCS with powerful programmable capabilities in two other subsystems that are functional nodes for processing requirements -- the Flight Data Subsystem (FDS) and the Attitude and Articulation Control Subsystem (AACS). Galileo architecture goes a step further with the distribution of microprocessors to major payload subsystems. Proliferation of local microprocessors can lead to excessive complexity just as a single centralized system can be over-constraining for subsystem requirements. Some middle solution of retaining a system level executive capability while delegating appropriate resources to processors sized for individual subsystems is probably most appropriate to a centralized or a decentralized architecture.

This topic describes a technique for incorporating the on-board data processing and control functions required for autonomous control and fault management into the set of possible spacecraft system processing and control architectures. This technique is readily adaptable to both 1) already developed spacecraft processing designs where the constraint of minimal change to the existing subsystem designs is levied, thereby forcing add-on design procedures, and 2) new spacecraft designs where the autonomy-related functions can be integrated into the new system and subsystem designs during the initial design phase. Furthermore, the techniques described herein can be used for multi-mission applications covering a wide spectrum of 1) processing requirements sophistication and 2) spacecraft processing and control architectures. Therefore, the technique may be used to incorporate fault management features into diversified spacecraft processing and control architectures covering the range from highly centralized to highly decentralized organization. The steps described in the subsequent subsections are chronological in order and include rationale to justify their applicability to the autonomous control and fault management design process. Furthermore, each step is assessed with respect to its use in two possible spacecraft processing and control applications representing comparative extremes in the spectrum of mission requirements and associated architectural characteristics. These reference applications are 1) a satellite representing an already-developed design currently having very little autonomy and extremely limited on-board computer capability and 2) a generic autonomous spacecraft design architecture having characteristics of a high level of autonomy and a highly distributed computer capability which are integrated into the initial design.

*By Wayne E. Arens

2.1.1.1 Baseline System Functional Requirements. The baseline functional requirements for the spacecraft form the initial set of needs that must be satisfied by the data processing and control architecture. The derivation of this set of baseline functions is described in Topic 1.1.1 of Part III of the Handbook. Mission function performance needs are emphasized, and health and welfare functions are clearly identified. The need for control and fault management is recognized in the provision of appropriate block and functional redundancy in the system. Subsystem level functions must recognize the need to provide a sufficient level of information so that fault diagnostics are available for autonomous and/or ground control.

Particular attention should be given to data processing requirements associated with command, telemetry (engineering and payload) and attitude control and navigation functions. From experience, the characteristics of these functions and their associated subsystems have placed major processing requirements on the spacecraft design. Meeting these baseline mission requirements is a major architectural consideration that must be satisfied along with autonomy requirements.

2.1.1.2 Identification of Autonomy Needs. The total system functional requirements discussed above and derived by Topic 1.1.1 of Part III are now compared with autonomy requirements to identify those functions that must be provided on-board. This process is described in Topic 1.1.2 of Part III. These functions will have data processing and control requirements that must be characterized for their impact upon system architecture design along with other mission requirements. The requirements imposed by these add-on functions are then translated into specific requirements for additional sensing, processing, redundancy, and interfaces at both the system and subsystem levels of the baseline design.

2.1.1.3 Autonomous Spacecraft System Design. It is assumed that some centralized executive-level computer processing and control service will be provided to all subsystems. This assumption is independent of the mission application and the spacecraft development mode. Provisions for accommodating additional sensors, processing, redundancy, and input/output interfaces are incorporated as required into the subsystem designs of the nonautonomous functional design evolving from 2.1.1.1 to satisfy the autonomy needs defined in 2.1.1.2. The specific design modifications involved depend upon 1) the mission requirements imposed upon the spacecraft, and 2) the spacecraft design mode, e.g., already developed spacecraft design versus new spacecraft design.

For an already developed spacecraft design, where minimal subsystem design change is an imposed constraint, a new subsystem is added to the nonautonomous spacecraft design to provide for the centralized computer executive-level control processes. This implies the possibility of more than one centralized executive-level computer.

For a new spacecraft design, only one centralized executive-level computer subsystem is incorporated. If the nonautonomous design evolving from 2.1.1.1 already includes a centralized executive-level computer for service function purposes, it may be modified to accommodate the control and fault management functions required for autonomy.

Fault management of subsystems that 1) do not require computer capability for performing their service functions, and 2) require only relatively simple fault management functions to be performed to achieve the required level of autonomy are accommodated by the centralized executive-level computer. In such instances, a subsystem is responsible for supplying all necessary sensor information and/or diagnostic responses via appropriate interfaces to the central executive. For such subsystems, the nonautonomous design evolving from 2.1.1.1 is not modified.

In contrast, when subsystems of a nonautonomous spacecraft design evolving from 2.1.1.1 require 1) additional sensors and/or 2) more computing capability than that available from the centralized executive, to achieve the level of autonomy required by the mission, additional sensors and/or computer processing capability are added to the subsystem design whether it is already developed or a new design. This subsystem computer performs fault management functions at the subsystem level under the high-level control of the central executive. It therefore must maintain an appropriate two-way communication interface with the central executive.

For already developed designs, such as described for the DSCS III spacecraft in Volume III of the SD-TR-81-87, additional processing capability is provided in the form of a separate add-on module designated as a Distributed Processing Unit (DPU). Subsystem sensor information is routed to the dedicated DPU which uses the information to perform fault detection and associated diagnostics for its assigned subsystem in support of the centralized executive-level computer. Such support provides processed fault diagnostic information to the centralized executive which is responsible for the fault recovery function of command generation and issuance, e.g., the DPU does not issue fault correction commands.

For a new spacecraft design where a subsystem requires dedicated computer capability to achieve the necessary level of autonomy, such capability is integrated into the internal subsystem design. If the design already requires a computer capability, it is simply modified to accommodate the required fault management functions. If it does not, a computer capability is added to accommodate the fault management functions. In either case, unlike the DPU application to already developed designs, in which fault recovery commands can come only from the centralized executive, subsystem-dedicated computers in new designs have the option of providing a full complement of fault management functions, including recovery from faults which are unique to the internal subsystem.

2.1.1.4 Centralized Executive Design. Using the autonomous spacecraft system design characteristics evolving from 2.1.1.3 as a basis, the health and welfare related functional and design requirements for the

centralized executive-level fault management computer functions are defined. Whether or not the centralized fault management computer capability 1) represents a separate add-on subsystem to accomplish the fault management functions for already developed spacecraft, or 2) is integrated, for new spacecraft designs, into a single executive-level subsystem that may perform other function in addition to fault management, the functional and design requirements are defined assuming the availability of at least the following hardware elements:

- (1) Central Processing Unit (CPU).
- (2) Read-Only Memory (ROM).
- (3) Random Access Memory (RAM).
- (4) Nonvolatile Memory (NVM).

A centralized executive architectural design, capable of providing self-test and self-repair of its constituent elements, is defined for incorporating the above functional elements in such a manner that all of the fault management functional and design requirements imposed upon the centralized executive can be accommodated. Using this architectural design for the centralized executive function, the software required for incorporating the necessary algorithms, to achieve 1) self-test and self-repair and 2) the fault management executive-level functions required by the specific mission application, is defined. Based upon the software requirements, design tradeoffs are performed to finalize the hardware computational and storage performance capability. Any additional fault management processing and control capability required by a specific mission, but not provided by the centralized executive because of practical limitations resulting from the design tradeoffs, is allocated for distribution to appropriate subsystems.

2.1.1.5 Distributed Processing Design. Using 1) the autonomous system design characteristics evolving from 2.1.1.3, and 2) the centralized executive design characteristics evolving from 2.1.1.4 as a basis, the health and welfare related functional and design requirements that must be distributed to specific subsystems to meet the mission autonomy needs evolving from 2.1.1.2 are defined. Whether or not this distributed processing capability comprises an add-on DPC as defined in 2.1.1.3 for already developed designs, or is integrated into a new subsystem design, the functional and design requirements are defined assuming the availability of at least the following hardware elements:

1. Central Processing Unit (CPU).
2. Read-Only Memory (ROM).

For already developed spacecraft designs, a separate DPU module, as defined in 2.1.1.3, is designed and added to an existing subsystem to accomplish the distributed processing needs required for fault-management of that subsystem. The DPU design includes signal conditioning circuitry to accommodate additional sensor information, provided by the subsystem, which is not included in the output telemetry data stream. Furthermore, the DPU is designed without a self-test and self-repair capability. A separate standby redundant DPU is provided in case of failure. Fault management functions for DPUs are performed by the centralized executive-level fault management computer described in 2.1.1.4.

For new spacecraft designs, the required fault management distributed processing capability for a given subsystem is integrated into the new subsystem design. Depending upon the specific mission needs, this capability may or may not issue fault recovery commands. For mission applications involving highly centralized spacecraft architectures for processing and control, all fault management commands will tend to be issued by the centralized executive in the same manner as for already developed designs using DPUs. For such cases, like a DPU, the subsystem fault management computer is designed without a self-test and self-repair capability. An example of a highly decentralized spacecraft processing and control architecture is described in Appendix D.

2.1.1.5 Common Memory Design. Using 1) the autonomous system characteristics evolving from 2.1.1.3, 2) the centralized executive characteristics evolving from 2.1.1.4, and 3) the distributed processing characteristics evolving from 2.1.1.5 as a basis, the functional and design requirements for a nonvolatile mass memory, capable of providing a mass storage repository for fault history audit trail data, critical system level software routines, and critical subsystem-level software subroutines is defined.

For already developed spacecraft designs, a new subsystem, such as the Data Memory Subsystem (DMS) described for the autonomous DSCS III design option in Volume III of SD-TR-81-87, is designed for addition to the spacecraft.

For new designs, the fault management storage requirements are incorporated into a Common Memory Subsystem (CMS) design that serves all mass storage requirements of the spacecraft. An example of such a CMS application is provided for the highly decentralized architecture described in Appendix D.

2.1.2* Reliability and Redundancy

The addition of autonomous operational capability increases the complexity of spacecraft design. The increased complexity provides the opportunity for a more robust design. To be successful, an attention is paid to the design of the autonomous operational capability. The design of the autonomous operational capability is a critical part of the spacecraft design. The design of the autonomous operational capability is a critical part of the spacecraft design. The design of the autonomous operational capability is a critical part of the spacecraft design.

2.1.2.1 Basic Approaches. Basic approaches that are of increased importance in the presence of autonomy goals include effective over-design when weight, space, and cost limitations permit, deratings of parts/devices, simplicity of design features, standardization to flight proven parts, devices, circuits materials/processes, a minimum number of total parts, a minimum number of device types in the design, and design providing for testability and inspectability.

Reliability engineering for autonomous spacecraft designs must support a high inherent reliability in the basic functional features of the design just as for non-autonomous designs. The result of reliability engineering analysis must also support the selection of functions that will require fault tolerance and fault recovery mechanizations to achieve the reliability required by the mission. These types of functions that are mission critical or which have a high design reliability payoff when supported by active autonomous fault tolerance should be identified as early in the design process as possible.

2.1.2.2 Function Relationship Analysis. Functional relationship analyses initiated early in the design program define sequences of related dependent functions extending from top level mission events through spacecraft actions, subsystems actions, and circuit functions, to part level functions. These analyses effectively provide early participative support to the design development process by identifying single failure points in the design at various levels, by indication of the frequency of function application in mission operations, by identifying areas requiring more extensive reliability engineering and design considerations, and by highlighting areas requiring extensive inspection and testing. All of these outputs are directly applicable to the early selection and development of fault tolerance, isolation, and recovery features required to meet the autonomy goals.

2.1.2.3 Failure Mode Effect Analyses. Failure mode effect analyses need to be function-oriented for the earliest and most direct application to support the design of the fault recovery and autonomy features. Function failure effect analyses should be accomplished at levels from subsystem functions to part level functions. Corresponding coverage of mechanical and electro-mechanical features should be provided by function failure tree techniques.

2.1.2.4 Redundancy Provisions. Redundancy provisions in functional designs are often used to increase the overall system reliability of the design and to significantly increase the expected operational life. Unfortunately, redundancy is sometimes considered as an alternative to effort required to achieve maximum reliability of a simple non-redundant design by effective application of basic reliability techniques. With the additional requirements of autonomy, redundancy must be considered only after achieving maximum practical reliability by the elimination of all practical sources of unreliability with non-redundant design.

A factor that must be considered, especially when extreme gains in reliability are desired, is that redundancy is not always pure gain; redundancy in any application is an increase in complexity that carries with it some cost in possible unreliability.

Applications of redundancy involve consideration of several characteristics. First, the level at which provisions are added to accomplish the function by a second, or multiple means, i.e., at spacecraft system level, at subsystem level, at circuit level, or at part/device level. Second, the scope, or extent of the redundancy which provides a redundant means for the point of initial application down to the part level, or backing-up only a portion, or portions of the overall functional sequence. Third, a decision whether the redundancy will be 'Active' (i.e., powered along with the primary mechanization) or be in 'Standby' status (i.e., inactive until failure of the primary). Several basic types of redundancy vary with respect to these characteristics.

2.1.2.4.1 Functional Redundancy. Functional redundancy involves providing more than one means (two or more depending on criticality and maximum possible inherent reliability of the individual links) of accomplishing a given function. The secondary mechanizations may encompass a completely or partially different and separate functional design approach, or it may be a duplicate of the primary. Variation in approach provides an avoidance of common fault modes.

2.1.2.4.2 Cooperative Redundancy. Cooperative redundancy involves splitting the equipment performing the function into two or more completely or partially independent portions such that critical elements can fail without terminating the total function; some degradation may be encountered. Mechanization is usually in 'Active' status to avoid the requirement for detection/activation.

2.1.2.4.3 Block Redundancy. Block redundancy provides two or more nominally identical elements which perform the same function. Piece parts in series or parallel arrangements, depending on the nature of the most probable part failure mode, and cross strapping of more complex devices/functional circuit assemblies are typical examples. Though usually set up in 'Active' state, detection/activation may be added to gain an increase in total expected operational life. Care is required in designing detection/activation to avoid expending the potential gains of redundancy on loss of reliability from increased complexity of the added functions.

2.1.3* Fault Management

Fault management is an active control response to the occurrence of a fault condition. Design and analysis for reliability constitute the first level of assurance that a given spacecraft function will operate properly. Provision for redundant implementation of a function allows a back-up to the reliability of the basic design. An autonomous spacecraft requires on-board control of redundancy or other repair mechanisms to utilize this backup capability. This fault management process consists of three logical steps:

- (1) Detection - Sensing the occurrences of a fault condition.
- (2) Isolation - Identifying the location of the fault and the appropriate response.
- (3) Correction - Reconfiguration of on-board resources to correct consequences of the fault.

The remainder of this topic discusses the characteristics of faults that influence the selection of implementation and techniques available for implementation of the detection, isolation, and correction steps.

2.1.3.1 Fault Characteristics. Characteristics of a defined fault are directly related to the resources required for fault management and the degree of protection afforded by the fault management process. The impact of a fault on spacecraft operations, the level of spacecraft architecture at which a fault is defined, and the interaction of the fault with spacecraft subsystems and operating modes are primary characteristics of concern.

2.1.3.1.1 Operational Impact of the Fault. The operational impact of a fault provides an important measure of the criticality of providing protection. It is a convenient ranking characteristic in trade-offs among other faults and in establishing a priority of response if several faults are detected or are being corrected simultaneously. In decreasing order of importance, potential impacts are:

- (1) Catastrophic loss of the spacecraft.
- (2) Complete loss of mission functions.
- (3) Partial loss or degradation of mission functions.
- (4) Loss or degradation of a subsystem function.
- (5) Loss of fault management or maintenance capability.
- (6) No significant impact.

*By P. R. Turner

The operational impact of a fault may vary with different mission phases or spacecraft operating modes. Failure of a sensor utilized only in a spin-stabilized mode of a three-axis stable spacecraft would not affect normal on-orbit operations, but might be important if a reacquisition of references was required.

2.1.3.1.2 Fault Definition Level. A fault may be defined as the interruption of service of a function at several different levels of the spacecraft functional hierarchy. Specific levels of definition are:

- (1) System Function Level - Examples of major system functions to be protected are attitude pointing and uplink command capability. Though these functions may be largely or entirely implemented in a specific spacecraft subsystem, their impact on spacecraft system operating modes and other subsystems relegates them to system level importance.
- (2) Subsystem Function Level - Fuel tank pressure indicator failure and power converter failures are examples of this level of fault.
- (3) Assembly Level - Failure of a single sensor element in an attitude reference sensor consisting of multiple detector elements represents a low level fault condition. The failure may be corrected at the subassembly level without impact on higher order functions if design of the assembly permits.

The level at which a fault is defined is important to providing the highest possible degree of fault protection with the minimum expenditure of autonomous control resources. Too low a level of fault definition results in additional definition of faults at a similar level with similar effects and resource requirements. Too high a level of definition may not allow a timely response to a fault or allow adequate isolation of the fault. Some factors that influence the level of fault definition are:

- (1) Available repair level for correction.
- (2) Time criticality of response to detected fault.
- (3) Availability of techniques for isolating the fault.
- (4) Level of functional architecture to which the fault can be isolated.

The Viking CMDLOS algorithm in Appendix C of this Handbook is an example of a high level fault definition with an example implementation. The Viking Pressurant Regulator Failure algorithm of Appendix C provides an example of a lower level fault whose critical nature demanded a fault management response at the assembly level.

2.1.3.1.3 External Interfaces. A specific defined fault has a range of potential interface impacts. Some possibilities are:

- (1) Payload operations impact.
- (2) Forced changes in spacecraft operating mode.
- (3) Reconfiguration of one or more external subsystems required.
- (4) Reconfiguration of subsystems containing fault required.
- (5) Redundancy switching with reconfiguration not required.
- (6) No direct impact outside of failed unit.

These characteristics may be related closely or loosely to the operational impact of the fault. A high degree of external interfaces implies more resources required for isolation and correction and the potential need for a spacecraft system level executive involvement in the implementation. No significant interfaces outside of subsystem favors a local subsystem level implementation of fault management.

2.1.3.2 Fault Detection. Fault detection consists of sensing the possible occurrence of a fault condition. Detection of a fault is usually considered the first step in the fault management process. It may also be considered as a final step in closing the loop of the autonomous process after the correction process is completed. The failure to detect the fault after repair action can be considered to verify that the fault management process has been successfully performed. The three primary methods by which this may be achieved are:

- (1) Direct measurement.
- (2) Indirect measurement of symptoms.
- (3) Inference.

2.1.3.2.1 Direct Measurement. Direct measurement techniques require the provision of a sensor that will give unambiguous indications of a specific fault. The sensor must be provided in the design process. This technique is most likely to be applied to highly visible faults with serious system impact and/or time critical response requirements.

2.1.3.2.2 Indirect Measurement. Indirect measurement utilizes the sensing of parameters that can be affected by the fault condition. Second order "symptoms" that point to the fault must be selected to be as unambiguous as possible. Ambiguous circumstances may be resolved by sensing two or more

"symptoms" and correlating them to assure detection. This technique is particularly applicable to "add on" fault management schemes where it is not feasible to add direct measurement sensors.

2.1.3.2.3 Inference of Fault Condition. Inference utilizes the occurrence or lack of occurrence of an event to indicate a potential fault. Failure to receive a valid command within a software specifiable length of time is considered sufficient evidence to infer an uplink fault in the Viking Command Loss algorithm of Appendix C. Inference provides a low resource means of implementing fault detection, but ambiguity can be a major shortcoming.

2.1.3.3 Fault Isolation. Isolation of a fault consist of identifying the proper response for corrective action. Isolation is trivial for a fault which is detected unambiguously and has only one appropriate corrective response. Conditions requiring more extensive isolation logic include:

- (1) Detected symptom is common to several faults.
- (2) Fault has more than one possible corrective action.
- (3) Fault may be transient or permanent.
- (4) Fault may be correlated with other faults.
- (5) Faults and/or corrective actions have different priorities.

Isolation logic will tend to be fault specific and is best implemented in software. Design of this logic is usually the most challenging part of fault management algorithm design. Fault correction response can frequently be implemented as stored command tables for sequential execution. The isolation logic must apply logical tests to available detection inputs and choose the appropriate correction command files.

2.1.3.3.1 Common Symptoms. A symptom common to several faults may be resolved by cross correlation with additional symptoms - positive or negative. The potential common faults should be examined in a priority order established by criticality of time response, impact on the spacecraft, or ease of resolution.

2.1.3.3.2 Multiple Corrective Response. A fault with more than one possible corrective response may be trivially isolated by only providing one response, probably the most critical. If this is not possible, some priority scheme must be provided to select a solution, test for success, and initiate the next most likely corrective response if the first was not successful. Such "trial and error" approaches have the shortcoming of not being responsive to time critical faults and requiring a large amount of commanding and reconfiguration. On the plus side, they may correct multiple faults and provide a series of

solutions for complex sets of faults that are difficult to diagnose and isolate directly from available data. The "Tree Switch" corrective response described in 2.1.3.4 is an example of this approach.

2.1.3.3.3 Transient/Permanent Fault Identification. Transient versus permanent fault identification can usually be achieved in the time domain. Logical processing of nominal operations should be tolerant to transient faults caused by noise, radiation environment, or other causes. If this is the case, a delayed response to one or more occurrences of a detected fault may be sufficient to differentiate those cases. Counts of fault detection incidents may be used to indicate the frequency of transient faults, which may be a symptom of a developing permanent fault. Persistence of the detected fault beyond a critical period may suffice to identify the condition. It may be necessary to correlate the detection indication with other data, however. A sun sensor that is not providing output could be operating properly if the spacecraft is in an eclipse condition or if its controlling electronics are commanded off.

2.1.3.3.4 Interrelated Faults. A fault which is correlated with other faults requires a predefined priority of response and may require correlation logic to ensure that the parent fault is corrected before attempting to correcting the secondary fault. The complexity of these conditions may be illustrated by an attitude control fault causing loss of solar array power and eventual thermal abnormalities. A further level of complexity arises if the attitude fault is caused by a thruster which is stuck open. The open thruster must be closed or isolated before attitude re-acquisition may be accomplished with subsequent restoration of solar array power and normal thermal conditions. Prioritization is simplified in a centralized control structure which must perform sequentially. Provision of decentralized control means that the elements of the decentralized structure must communicate information regarding correlated faults to establish a unified response before initiating corrective actions. Presence of several faults can serve to inhibit response to secondary faults until the more serious parent faults are isolated and corrected.

2.1.3.3.5 Correction Priorities. Uncorrelated faults with different correction priorities are a problem if their correction responses compete for control or service resources. The problem is simplified for centralized control architectures, as only one can be treated at a time and a priority scheme must be provided by an executive. Decentralized architectures present a more subtle difficulty. The control resources may be available to attempt to correct all faults simultaneously, but they may have conflicting requirements for spacecraft power or operating mode configurations. Faults involving loss of power generation capability and thermal imbalance are one such combination. Load shedding is a standard response to power problems, but thermal changes may require the powering of active heaters. These conflicting correction requirements must be identified and addressed by isolation logic.

2.1.3.4 Fault Correction. The principal fault correction response is to activate redundant block or functional capability to replace a failed element. This involves issuing of a predetermined sequence of commands or logically selecting among blocks of predetermined commands for specific fault conditions. The actual correction occurs as hardware state changes. These state changes may be hardware implemented for simple faults or those which require an immediate response and can be unambiguously detected and isolated. The majority of complex faults, however, require correction responses to be commanded through programmable software logic enabling table driven command sequences.

The complexity of the correction response to a fault may involve any or all of the following factors:

- (1) Isolation level of the fault.
- (2) Redundancy level provided.
- (3) Redundancy type.
- (4) Fault interface impacts.

2.1.3.4.1 Simplification of Correction Strategy. Correction strategy is simplified if the detection and isolation steps can directly locate the faulty element. If this is not the case, correction actions may have to proceed in a "trial and error" fashion until the fault is no longer detected or takes a worst case of action that replaces non-faulty elements in addition to the faulty one. The easiest case is usually found for easily identified faults with major impacts on spacecraft safety. Design forethought is necessary to provide the appropriate detection and isolation characteristics for the potential fault. The more complex and non-optimal responses usually arise when minimal control resources are available and/or the fault is not identified early in the design process.

"Tree switching" until a fault is corrected is an approach used for the Viking Command Loss algorithm (Appendix C). This technique consists of providing a table of uplink redundant element that are alternatively switched from one redundant block to the other in sequence until the fault condition terminates. This technique provides a generalized response to a fault that is isolated at a very high level (the uplink function) with minimal resources for detection and isolation (provision for a counter, reset of the counter, and a test). It is capable of responding to a wide variety of faults that may result in loss of uplink capability and can correct multiple faults as long as no two redundant elements of a function are both lost. The basic cost is in response time and the amount of subsystem reconfiguration required.

2.1.3.4.2 Redundant Element Availability. The available level of the redundant elements used to correct a fault has an important impact on the required corrective response. A low level of redundancy may allow repair with minimum impact on spacecraft operating modes and other subsystems. It may,

however, require a significant increase in overhead for detection and isolation if a low level of redundancy is provided across the design without consideration of the reliability and criticality characteristics of the elements.

2.1.3.4.3 Block and Functional Redundancy. Block and functional redundancy may have different correction characteristics. Ideally, block redundancy should be the easiest to control at any level. The faulty element is turned off and the redundant element is substituted. Any additional complexity is due to the interface characteristics of the fault. Functional redundancy may require little corrective action, depending on whether the functionally redundant elements are in use for other purposes or as a powered backup. A functional redundancy that normally operates to fulfill a function unrelated to the faulty element may only affect the processing logic of the faulty function.

2.1.3.4.4 Fault Impact Across Interfaces. A major impact upon correction requirements is the impact of the fault on other spacecraft subsystems and functions. Correction action that requires a change in the spacecraft operating mode or a reconfiguration of several subsystems must be constrained by the operating and fault characteristics of those impacted subsystems. Such faults will require extensive commanding and have a larger memory storage requirement than faults with little or no outside impact.

2.1.4* Software Fault Detection Implementation

Implementation of an autonomous control structure involves the use of hardware, firmware, and software. Figure III-7 indicates the logical relationship between these implementations. Hardware logic and firmware logic burned into Read-Only Memory (ROM) are impossible or awkward to change after launch. Reprogrammable software allows the maximum flexibility for changes in control logic with different mission phases and changing spacecraft health conditions, as well as allowing improvements to control implementation with increased operational experience. Increases in spacecraft complexity will drive a need for the flexibility provided by software control logic. This topic will address control logic techniques that may be utilized for firmware as well as software, however, the term software shall be used to refer to the expected mode of implementation.

2.1.4.1 Software and Faults. Software routines serve as a means for providing fault management logic and as a source of potential faults themselves. Fault-tolerant computer and processor designs can provide a great deal of protection against faults within the logical engine itself, but there remains the prospect of external problems that provide improper input to software logic without triggering any of the intended protection. Such occurrences can cause the software logic to operate in a faulty manner or to produce outputs which serve to propagate the fault, usually making it more difficult to trace its origin. The techniques discussed in this topic are more properly considered as techniques for fault detection rather

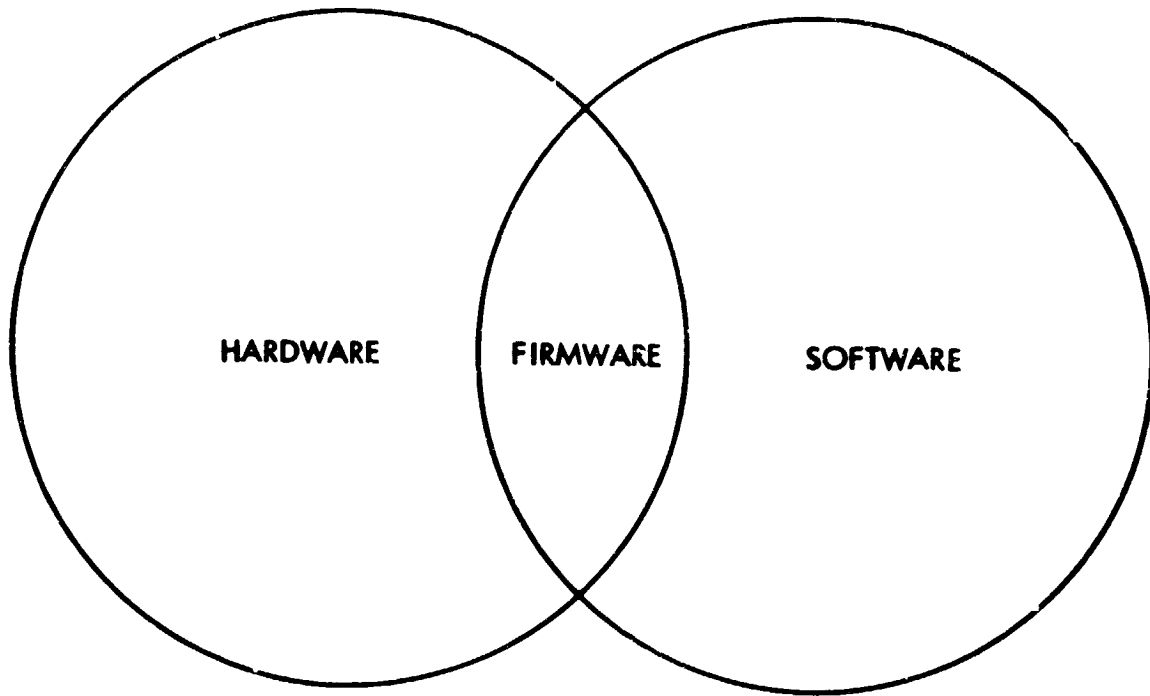


Figure III-7. Relationship Among Logic Implementations

than fault management. The isolation and action response to a fault indication are highly dependent upon the fault. It is easier to enumerate generic techniques for determining the possibility of a fault than responses. Or rather, the response can be summarized as do something or do nothing. The details of these choices are left to the ingenuity of the implementor.

Software logic processing can be characterized by three prerequisite phases of inputs, processing, and outputs. The prospects for fault detection lie in prevention of processing faulty inputs, noting faulty results in processing, or preventing the output of faulty products. The best placement of protective logic is highly dependent upon the hardware and software characteristics of the function protected. Fault management logic is an overhead that utilizes resources which could be devoted to direct control logic. Consequently, the identification of functions protected and the system or subsystem implementation of protection should be a major system design consideration to prevent waste of memory resources.

Software processed digital information falls into one of two categories: bilevel or discrete digital words. A bilevel is represented as a single digital bit which relates one of two possible states. Bilevel data types are the simplest possible type of logical fault indicator. They do not, however, have any way of indicating their own faulty operation unless correlated with another indicator. This correlation must be provided through some sort of discrete word structure or relations to one or more bilevels and their associated fault behavior. Discrete digital words provide more scope for detection of faulty content. Discrete words, singly or in groups, form software instructions in memory, data to be processed or output and command structures for external devices.

2.1.4.2 Format Protocol Techniques. Protocols which define characteristics of valid digital words are particularly useful in input/output checks. Transient or permanent faults can affect analog/digital conversations, register loads, data bus communications, and memory contents. Format protocols may be implemented with or without hardware support to protect against many of these faults. Complex error detecting and correcting code techniques have been developed for high bandwidth applications, but these will not be addressed in this topic. The emphasis is on simple techniques that can be applied at the low data rates typical of spacecraft control applications and which require a minimum of software overhead.

2.1.4.2.1 Parity Checks. Setting an additional bit of a data word to consistently provide odd or even parity of the set bits is a time honored technique for detecting many single or multiple bit errors. It is usually applied in hardwired computer and data transfer hardware, but may be checked by software as well.

2.1.4.2.2 Sparse Word Bilevel. Consistency checking of a bilevel indication may be provided by utilizing only one bit in a word at a time. An eight-bit word will only signify eight distinct conditions with this technique. The basic error test is that only one of the eight bits in the word may be set at a time. If more than one bit appears set, an error has occurred. This technique can be applied to indicate conditions which are processed sequentially or are mutually exclusive. This technique is used to monitor Voyager AACS operating mode commands, as only one mode should be active or commanded at any time.

2.1.4.2.3 First and Last Bits Identical. A protocol requiring the first and last bits of a command word to be identically one or zero is used in the Voyager AACS to protect against hardware register "fill" errors. This redundant bit technique is used in parallel with a parity check on the nonredundant bits to provide additional protection against bit errors.

2.1.4.2.4 Sequence of Protocols. Requiring two separate protocols to occur in specific sequence is useful in protecting critical data and command actions. Commands can require a precursor to enable execution or storage. The precursor may be functional, or only serve as an enable for the critical action. Data transfers may be protected by supplying starting and ending addresses for the transfer in sequence with separate protocols. As an example, the start address might occur first with first and last bits zero, followed immediately by the end address with first and last bits set to one. Both techniques are utilized in Voyager AACS along with a class of commands that must be preceded by a data transmission to be valid.

2.1.4.2.5 Mode Validity Check. Command or data transmission may be valid only in a specific operating or processing mode. Execution of some types of thruster commands may only be valid for a specific AACS operating mode and should be inhibited for others. Similarly, some operating modes of a subsystem may not be entered from others. Use of sparse word bilevels to indicate current and now commanded modes can increase protection against bit errors in such a validity check.

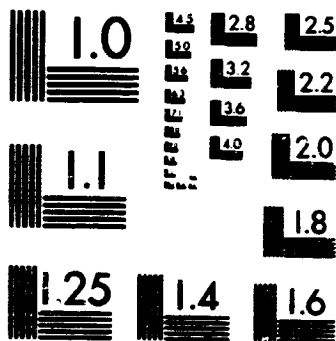
2.1.4.2.6 Heartbeat. Heartbeat is a means of verifying the proper operation of a continuously cycling processor by reprogramming it to output a specified bit pattern at regular intervals. Cessation of the heartbeat signals the occurrence of a malfunction to any external processor that is monitoring the heartbeat. Protocol checks built into the heartbeat pattern may be used to verify the proper operation of the communications bus between processors. This method was used on Voyager to allow the CCS to monitor the health of the AACS. Some internal AACS fault detection routines signalled the CCS of a need to switch to redundant AACS hardware by shutting off the heartbeat.

2.1.4.3 Performance Assessment Techniques. Techniques for assessing the proper performance of an autonomous operation can be applied to input, output, or processing. Routines to implement command processes or telemetry tend to primarily involve data handling and logic tests. Checks on input and output

83

6920

3 1/2 B



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

may suffice to detect errors that pass format protocols. Navigation and attitude control involve significant amounts of complex computation with logic paths and final results dependent upon the outcome. Additional techniques are required to locate faulty operation in a routine before vital evidence of the problem is destroyed and to assess the operation of the software logic. These techniques may be applied to these ends.

2.1.4.3.1 Magnitude Limit Checks. Comparison of the value of a quantity with known or empirically selected bounds is applicable to inputs, outputs, and computed quantities. Flight experience has suggested that wide tolerances be initially selected for dynamic quantities. As experience is gained with noise levels, transient faults, and subsystem interactions, the limits may be tightened with less likelihood of a false triggering of fault detection.

2.1.4.3.2 Sign Checks. A quantity that should always be positive or negative can be tested for consistency.

2.1.4.3.3 Evaluation of Constants. Natural constants involved in computation of physical processes may be evaluated from current measured or computed values. Navigation may use gravitational constants to check for gross errors in estimated or propagated state. An accurate navigation system could utilize the technique to detect thruster leaks or performance degradation. The applicability of the technique is dependent upon the accuracy with which the constant can be evaluated and the implications of the accuracy on the detection of the fault.

2.1.4.3.4 Overflow Test. Computations involving summing inputs from a variety of sources are prone to overflow. Such computations occur frequently in control law evaluations, and an overflow might produce faulty control output or characteristic behavior of a saturated subsystem component. In either case, it is important to protect the output of the computations under these conditions.

2.1.4.3.5 Correlation of Inputs. Comparison of inputs from redundant sources can be used to identify faulty data. The Voyager AACS gyro package design provides redundant measurements of rates about each axis. Comparisons are used to initiate fault isolation responses when the inputs differ significantly. Comparisons that differ intermittently may be a result of noise or impending failure. The immediate response to a single occurrence could be to ignore the input, with a more serious response taken if the symptom continues.

2.1.4.3.6 Redundant Computation. Critical quantities may be computed in series or parallel as a validation of performance. Input data from separate sources or separate processing algorithms may be used with the same data. The applicability of this technique is dependent upon system architecture or the availability of alternative computation algorithms.

2.1.4.3.7 Checksums. Software may be used to implement a checksum computation on memory contents that have not been modified or which should remain constant. This provides assurance that changes or faults have not occurred in blocks of input data or coded logic. This might be an appropriate technique for use by an executive routine when software has been reloaded or a data message has been stored in memory.

2.1.4.4 Logical Fault Detection Techniques. These techniques are useful in detecting or diagnosing faults that occur in processing logic. The logical fault may be due to faulty data that was not detected, unusual characteristics of subsystem behavior, numerical instability in a computational algorithm, or a logical fallacy in the basic algorithm design. The source of the problem is not as important as the detection of it.

2.1.4.4.1 Execution Timing. The execution of a routine or block of quantifiable logic may start a timer which runs during the execution. The timer may have some a priori upper limit that should not be exceeded, or the timer could be examined after completion of execution for suspiciously long or short duration of execution. This technique may be useful as a high level check on major functional processing or a check on a lower level block of logic containing loops of indeterminate duration. A well-known nonvariable logic process could be checked to insure that it ran the proper amount of time. The "time out" feature can prevent endless loops from locking up a computer due to an error.

2.1.4.4.2 Loop Counter Limits. Counters can be placed in an interactive loop to determine the number of execution cycles. An a priori limit on the allowable number of cycles can be used to prevent infinite or excessive looping. This can be particularly valuable in loops that are to be executed until a computed estimate converges to within a tolerance of a final value. Such loops can rarely be tested over a full range of inputs and are frequently subject to instabilities.

2.1.4.4.3 Error Count Thresholds. Non-fatal errors that can be ignored if caused by transients should be tallied to provide a measure of their frequency of occurrence. This count may be useful only as a diagnostic, or may serve as an indication of an intermittent fault. An error count threshold can be set to trigger further fault diagnosis or corrective action if an excessive accumulation of such errors occurs.

2.1.4.4.4 Control Action Counts. A number of occurrences of a specific control action, such as thruster firing or momentum wheel unloads, can be indicative of a fault. Such indicators may be outside the responsibility of a subsystem level fault management scheme by virtue of their subtle interaction with the subsystem or the fact that external faults drive them through proper response of the subsystem. The major subsystem impact might be excessive use of subsystem resources (propellant, computation time, etc.).

2.1.4.4.5 Self-Test Algorithms. A system or subsystem level resource may contain a test routine which produces deterministic actual output for a specific input. This routine can be executed upon indication of certain faults or may be periodically executed as a routine action. A comparison of actual output with the expected output may be used as a fault indication or to provide diagnostic data for analysis of the logical performance of the system.

2.1.4.4.6 Ticket Checks. A ticket check is a means of tracing the logical flow in a routine's execution. Completion of or entry into a logic path is recorded by setting a bit in a "ticket word". The bit status of the word acts as a record of the progress of execution of the routine. Routines with fixed logic should have a deterministic "ticket" value after proper execution. Routines with variable logic may have a subset of acceptable ticket values or certain acceptable sequences of ticket values at different points in the logic flow. A major value of this technique is that it indicates what logic flow was taken in unacceptable cases and serves as a diagnostic tool as well as a check on proper sequential execution of logic.

2.1.5* Algorithm Development

The algorithms presented in Topic 2.1.6 were developed to augment hardware reliability and fault protection which was designed as part of the Viking and Voyager spacecraft subsystems. The development process which produced these supplementary software algorithms is summarized in this section. The description of the development process has been changed to incorporate lessons based on this previous experience and to present a development approach which accommodates the broader requirements of more fully autonomous systems than have flown to date. Emphasis has been placed on fault management techniques, rather than maintenance functions, because of the broader knowledge base in fault protection developed to date. The same basic algorithm development approach can be readily adapted to maintenance functions for fully autonomous systems, as was demonstrated during the Viking Extended Mission.

2.1.5.1 Algorithm Function. The algorithms contain spacecraft software logic whose purpose is to augment fault protection features incorporated in spacecraft hardware. Development of such software requires an intimate understanding of how the hardware functions in both normal and abnormal operation. The algorithm functions complement the hardware design and operating characteristics to achieve the required spacecraft fault protection or maintenance requirement. The optimum design process thus requires that hardware design, component selection, and algorithm function selection proceed simultaneously and with close coordination and a high degree of interaction. This process ensures that hardware and software functions are complementary and achieve the required degree of in-flight autonomy with minimum weight, power, cost, and complexity.

Note that the process just described is different from design practices to date (including those of Viking and Voyager) which have almost always added software algorithms for fault protection and maintenance to hardware which was already designed. This experience has suggested that early integration

*By Robert W. Rowley

of software function selection and design with the hardware design process would have produced a somewhat different design in both areas and would have enhanced the flight performance significantly.

Criteria for partitioning functions between hardware and software are summarized in 2.1.5.4. Flight experience is summarized in Part V.

2.1.5.2 Requirements. Fault protection requirements can be summarized as follows:

Maintain Commandability: The spacecraft must be able to receive and correctly process commands. This preserves the ability of ground controllers to assist in fault correction and sequencing the spacecraft.

Earth Point: The spacecraft must maintain a primary antenna pointed at earth. This is required to maintain commandability (uplink) and data return (downlink).

Preserve Power: The spacecraft must maintain a positive power margin under all operating conditions, including during fault correction activities.

Minimize consumable use: The spacecraft must minimize use of attitude control and trajectory correction propellant to maximize operating lifetime.

Note that fault protection requirements are not stated as software requirements. The software algorithms added to the spacecraft respond to hardware needs in achieving these overall system requirements.

Fault protection requirements for military spacecraft will include those described here for planetary spacecraft, but must be specified in the highest level project documents at the outset of the project, and must be reflected in lower level design documents which control the hardware/software system and subsystem design.

2.1.5.3 Critical Fault Selection. The following guidelines should govern the selection of the critical faults to be protected against.

- (1) A failure mode and effects analysis (FMEA) should be developed early and used to change hardware and software designs to minimize critical faults. (Current spacecraft design practice does not normally require an FMEA early enough to achieve an optimum balance of hardware and software fault protection).
- (2) The faults to be protected against should be those that affect spacecraft function. Project level requirements, including those included in 2.1.5.2 should determine the functions to be protected.

- (3) Faults should be grouped so that critical faults can be corrected with as little delay as possible. This not only preserves operability, but better prepares the spacecraft to protect itself against subsequent faults.
- (4) A corollary to the previous guideline is that critical faults should be isolated from the system to avoid later, perhaps unnecessary, switching of related elements.

2.1.5.4 Hardware/Software Tradeoffs. As discussed in 2.1.5.1, the hardware and software for autonomous systems must be developed in parallel, with fault protection applied in hardware, software, or both to best achieve mission requirements. This requires a significant change in current spacecraft development practice, where the hardware is usually well into design before software development begins. The results of current practice as applied to the Voyager Attitude and Articulation Control Subsystem has been summarized in Reference 1. This represents a broad assessment of fault protection processes and flight experience and forms much of the basis for current spacecraft fault protection design.

The following guidelines summarize design practices and tradeoff criteria which can influence the split between implementing fault protection in hardware and/or software.

- (1) Uncorrected faults often tend to propagate outside the failure area (e.g., excessive power use in one area may cause thermal problems in adjacent areas). Thorough modeling of system operation and rapid isolation of faulted elements is required.
- (2) Cross-strapped and redundant hardware elements (i.e., switchable elements) should be similar in size or function and as small as practical. Switchable elements often should be smaller than required to meet basic fault protection criteria such as single point failure protection. This enhances rapid detection and isolation of faults.
- (3) Faults should be detected by measuring what is required; inference and analytical techniques should only be used where direct measurement is impractical. Control algorithms and fault protection algorithms should be developed in parallel since they will be interactive.
- (4) Excessively tight margins can make fault management techniques more complex. Examples include power margins, fuse sizing, temperature margins and telecommunication link margins. Excessive activation of fault protection features with tight performance margins can propagate and cause additional problems.

- (5) The use of expected results (feed forward) from control laws and similar algorithms rather than commanded results to determine existence of a fault can reduce the number of false alarms. Such techniques require a thorough understanding of system performance and may be expensive in software, but enhance operability.
- (6) The selected computer architecture influences the timing, complexity, amount of interfaces required, and many other features of fault protection algorithms. Architecture selection should not neglect the requirements of fault protection (another reason why fault protection should be included early in system design rather than added on).

2.1.5.5 Algorithm Development Process. Selection of specific algorithms required to support spacecraft operation is an output of the spacecraft design process. Following the selection of the software functions to be developed, the following steps summarize the algorithm development process:

- (1) Conceptual Design - Although currently passing out of vogue, the flowchart remains a commonly used tool for summarizing the design and operation of algorithms (see, for example, Appendices B and C). They continue to remain invaluable not only in developing the initial design, but in visualizing how the algorithm functions during the detailed coding and development, during test and troubleshooting, and particularly during the reviews and critiquing sessions necessary to ensure a well thought-out design.

The design of fault protection software should be performed by personnel completely familiar with the design and operating characteristics of the hardware. Extensive test and analysis of operation under all conditions may be required to model performance under off-nominal conditions (such as thermal or voltage extremes) to ensure proper software responses to false triggers. Because hardware performance and idiosyncrasies must be well understood by the algorithm designers, the software logic design is best performed by the hardware cognizant organization.

- (2) Audit Trail - The primary purpose of the audit trail is to determine whether the fault protection features function properly. Algorithms must provide time-tagged information on significant events for inclusion in a master audit trail. To avoid excessive data accumulation, this time-tagged output must be in high-density, compressed format. However, it must contain adequate diagnostics, including timing, to reconstruct the cause and effect of each event in what can be a lengthy chain of fault protection actions.

As discussed in Reference 1, the audit trail provided in the Voyager design proved to be inadequate in diagnosing the experiences associated with the launch of Voyager 1. The result was a tedious ground reconstruction process. A conclusion from this experience is that proper audit trail design must be part of the learning process during algorithm design, development, and validation steps and is heavily influenced by experience gained during the development process.

- (3) Algorithm Development and Review - Coding, test, and review form an iterative process which must be repeated several times to ensure proper operation of the algorithm. Periodic reviews and demonstration (both formal and informal) of algorithm functioning are invaluable, with the informal peer group "what-if" discussions generally proving most useful. The purpose of the test and review sessions should be to uncover conditions which cause the algorithm to fail, either by not functioning properly, failing to function at all, or triggering when it shouldn't.

Software simulators should be used not only to verify normal operation, but to check operation in the presence of parity errors, noise, a complete spectrum of faults, and human operational errors.

Review and test should include the experience gained on previous programs as well as prior history with the algorithm of interest. Continuity of personnel is vital to successful development of autonomous flight software.

- (4) Validation - A formal test and validation phase should be performed at both the algorithm or subsystem level and at the spacecraft system level. Each algorithm should be tested at all assembly levels, since the level of interaction (and the potential for unexpected responses) increases. Integrated hardware tests have proven to be more helpful than pure software testing or simulations. After assuring normal operation, the software must be exposed to as many operating conditions as possible to flush out unexpected or wrong responses. Included in this matrix should be the introduction of off-normal and completely incorrect inputs to ensure that under no conditions will the software provoke responses other than those which achieve the original requirements.

REFERENCES

1. Rasmussen, R. D., and Litty, E. C., "A Voyager Attitude Control Perspective on Fault Tolerant Systems", AIAA Paper No. 81-1812, August 1981.

2.1.6* Autonomous Control and Fault Management Algorithms

The fault management algorithms used on recent planetary spacecraft illustrate the application of the fault management tools presented in this Handbook. While the availability of programmable on-board resources has enabled increasing fault protection over earlier hardwired designs, these efforts have necessarily been limited by on-board memory size to those faults considered most threatening to spacecraft health and to the ability of ground personnel to maintain communication. Except for a few critical events, such as launch or planetary orbit insertion, it has generally been acceptable for the fault routines to place the spacecraft in a safe mode awaiting ground action. Thus the fault protection software developed to date, while illustrating many of the principles discussed, is a relatively small portion of the software required by a complex autonomous spacecraft of Level 5 or better which must operate unattended for up to 6 months.

The algorithms provide a tier of reliability protection to back up the basic spacecraft design. Many algorithms serve to back up commanded turn, scan platform slew, or trajectory correction maneuvers which are executed under on-board control. Limit checks of sensor outputs and computer interface protocols are used to trigger many algorithms. Algorithms are not implicitly designed to check for secondary failures in the sensing of a fault or to account for a failure in their own logic design. Incidents such as these have been considered multiple failures and the attendant risk is considered in algorithm design and validation. Except for the Viking extended mission (discussed below), fault management algorithms are designed prior to launch. Extensive reviews, checkouts on software simulators, and validation during spacecraft system test are then performed. Nevertheless, experience with the spacecraft after launch frequently results in modifications to the algorithms. Changing spacecraft operation due to faults or end-of-life wearout failures also required post-launch software updates. The ability to modify the software in flight thus provides an invaluable tool to adapt spacecraft operation to changing conditions.

To aid in bridging the gap between fault protection algorithms developed to date and more extensive algorithms required for future spacecraft, a series of generalized algorithms have been developed as examples. The following sections summarize both these generic routines and the specific routines developed and flown on Viking Orbiter and Voyager.

2.1.6.1 Generalized Algorithm Forms. Based on a planetary spacecraft experience and the DSCS III assessment study. A series of generic algorithms has been developed to demonstrate techniques for detecting, isolating and correcting representative faults. Algorithm development proceeded to the level of functional flow charts and a general description of the processing required. No software development was attempted. A requirement of 60 days/6 months operation without ground support at Level 5 autonomy, as discussed in the SD-TR-81-87 was the primary driver in formulating the techniques developed. No specific spacecraft or mission was assumed. These algorithms are thus intended as generic examples of fault protection approaches rather than mission specific fault routines as presented in the following section.

*By R. W. Rowley

Algorithms were developed for power, attitude control, TT&C and propulsion functions. Table III-3 summarizes these algorithms. Several representative examples are included as Appendix B.

2.1.6.2 Specific Algorithms from Planetary Designs. The unique requirements of planetary exploration missions have forced the development of increasingly more complex fault protection routines for planetary spacecraft. Recent experience with the Viking Orbiter missions to Mars and the Voyager missions to Jupiter and Saturn has been documented and is presented in this section and in Appendix C.

Planetary mission requirements which force increasing application of on-board fault protection include long communication times, long mission durations and limited availability of deep-space tracking net time. For example, the Voyager prime mission (through Saturn encounter) was four years in duration with a two-way light time at Saturn of 2 hrs. 53 min. Tracking was often limited to one pass per day with 12 to 16 hours between passes. Critical planetary encounter sequences typically occur near the end of the spacecraft design life when failed or degraded elements are present and when communication times are longest.

Fault routines were developed after specific faults were identified by analysis, test or flight experience. Since fault protection by on-board software was not initially a project requirement, the ability to cope with the identified faults often depended on the flexibility already incorporated in the hardware design. Thus, the listings of Viking and Voyager algorithms which follow represent only part of the software control to be included in a fully autonomous spacecraft.

2.1.6.2.1 Viking Orbiter. The Viking Orbiter Computer Command Subsystem (CCS) performed spacecraft sequence control and contained all fault protection software. (The orbiter is described in more detail in Appendix A.) The spacecraft design used block redundancy extensively so that fault routines in many cases relied on switching of redundant elements. These algorithms shown in Table III-4, were designed to ensure completion of mission critical events such as Mars Orbit Insertion (MOI), maintain a command link in the event of a receiver failure or inability to process commands, maintain sun acquisition and downlink, and maintain spacecraft power. Except for critical mission phases such as MOI, the spacecraft usually reverted to a safe condition after exercising preprogrammed correction algorithms.

Following the primary mission, a lengthy extended mission was conducted until both orbiters ran out of attitude control gas. The CCS on each spacecraft was reprogrammed with additional control and fault management routines. These routines were designed to ease the workload on a greatly reduced ground operations team by performing routine operations such as autonomously charging batteries after occultations, and handling degraded operation as various wearout or end-of-life phenomena developed (such as attitude control gas jet leaks). These additional routines resulted in significantly more autonomous operation than had been designed into the original mission.

TABLE III-3
Generic Fault Management Algorithms

NAME	FUNCTION
Uplink Maintenance	Maintains a continuous telecommunications uplink by detecting and isolating anomalies in the receiving chain and either adjusting or replacing elements to restore performance.
Downlink Maintenance	Maintains a continuous telecommunications downlink by detecting and isolating anomalies in the transmitting chain and either adjusting or replacing elements to restore performance.
Load Fault Management	Monitors electrical load impedance and corrects a noncatastrophic fault condition. Action taken depends on criticality of load.
Battery Load Management	Provides an autonomous ability to manage power loads to prevent battery depth of discharge from reaching a predetermined critical point.
Computer Processor/ Memory Checkout	Detects and corrects faults at the block redundancy level in processor and memory elements of the spacecraft computer subsystem.
Telemetry Checkout	Detects and corrects faults at the hardware block redundancy level in the information acquisition and telemetry generation elements of the spacecraft telemetry subsystem.
Loss of References and Reacquisition	Maintains 3-axis acquisition of celestial references and provides fault management for fine sensors.
Attitude Control System Health Monitoring and Fault Protection	Monitors the health of ACS devices, detects and verifies faults, and recovers by switching to a redundant element or a back-up operating mode.
Thruster Management	Verifies proper thruster operations, reconditions a faulty thruster if possible, and replaces a failed thruster with a standby unit if degraded operation is not allowed.

TABLE III-4

Viking Orbiter Spacecraft Fault Management Algorithms - Primary Mission

NAME	FUNCTION
CCS Errors (ERROR)	Responded to anomalous CCS hardware or software conditions. Normally placed the CCS in a "wait" state (except during Mars Orbit Insertion maneuver).
Mars Orbit Insertion Power Transient (MOIMAU)	Provided a means to continue CCS execution of the Mars orbit insertion maneuver in the presence of a spacecraft power transient or attitude control electronics power changeover.
RF Power Loss (RFLOSS)	Corrected a low power output of either the exciter or TWT by cycling through all possible S-Band exciter/TWT combinations until the downlink was re-established.
Command Loss (CMDLOS)	Assumed a spacecraft failure if a command was not processed in a specified number of hours. Systematically switched redundant element until a valid command was received by the CCS.
Roll Reference Loss	Responded to a loss of Canopus reference star by commanding a flyback and sweep of the Canopus tracker instantaneous field of view to search for the star within the tracker's field of view followed by a roll of the spacecraft to search for the star.
ACE Power Changeover	Caused a switch to the redundant Attitude Control Electronics (ACE) under specific fault conditions.
Battery Charger Disconnect (BCHGDS)	Monitored the temperature of each of the two batteries during charging. Disconnected a battery charger from its respective battery if an over-temperature condition was detected.
Share Mode (SHRMOD)	Determined that the spacecraft was in a share mode and shed pre-assigned loads to allow the boost converter to boost the solar array voltage to the higher operating point.
Pressurant Regulator Failure (PRSREG)	Detected a propulsion regulator leak and isolated the regulator from the high pressure helium supply before the propellant tank relief valves could actuate.

The algorithms developed during the extended mission are summarized in Table III-5. Representative examples are documented in more detail in Appendix C.

2.1.6.2.2 Voyager. The Voyager spacecraft contains two computers which perform fault management activities, the Computer Command Subsystem (CCS) and the Attitude and Articulation Control Subsystem (AACS). A third computer, the Flight Data Subsystem (FDS), is not normally active in fault management. (The Voyager spacecraft is described in more detail in Appendix A). The AACS provides fault management for attitude control functions while the CCS provides fault management for the remainder of the spacecraft as well as serving as the spacecraft executive. These executive functions include fault checks on the AACS and AACS-CCS interfaces.

The spacecraft design used block redundancy extensively with the result that many fault correction algorithms rely on switching redundant elements to alleviate a problem. High priority was placed on maintaining earth pointing and a downlink and maintaining command capability. Algorithm development was guided by the overall Voyager reliability requirement that no single failure result in the loss of more than 50% of the engineering data or the loss of data from more than one science instrument. Additional goals of fault management include ensuring spacecraft health by managing power, minimizing expenditure of consumables (hydrazine), and checking the internal operation of the CCS and the AACS.

The Voyager fault routines are summarized in Table III-6. Representative examples are documented in detail in Appendix C.

TABLE III-5

Viking Orbiter Spacecraft Fault Management Algorithms - Extended Mission

NAME	FUNCTION
Battery Discharge Monitor (BATMON)	Monitored discharge current of the two batteries during occultation and configured the spacecraft to a safe state if state-of-charge was below a safe level due to loss of one battery.
Autonomous Battery Charging (BATCHARGE)	Autonomously recharged the batteries after solar occultation by monitoring battery temperature.
Science Power On (SINPON)	Prevented damage to science instruments from power transient at turn on by monitoring current. The instrument was turned off if an over-current was detected.
Receiver Switch (RCVRSW)	Protected against a receiver failure during extended occultation by monitoring oscillator current and switching to the back-up receiver if current fell below a preset level.
Downlink Off (DLOFF)	Insured that the spacecraft transmitter would be turned off at end of mission in the event of loss of command capability.
Accelerometer Monitor (ACLMON)	Terminated the propulsion burn-to-depletion test performed at the end of the mission by monitoring the accelerometer count and commanding engine shutdown when spacecraft acceleration dropped a preset amount.
Automatic Leak Clearing (CORKER)	Monitored position error to detect gas jet leaks and caused jets to actuate to attempt to clear the leak.
Stray Light (STRAY)	Monitored the Canopus star tracker and initiated sequences to prevent damage to the tracker and acquire or maintain star reference after stray light exposure.
Low-Rate Engineering Telemetry (DECOM)	Extracted selected data from the telemetry stream for use in fault protection algorithms.

TABLE III-6 (Sheet 1 of 2)

Voyager Spacecraft Fault Management Algorithms

NAME	FUNCTION
Command Loss (CMDLOS)	Corrected a failure to receive ground commands. A failure was assumed whenever a preset number of hours had elapsed since the last valid command received.
Radio Frequency Loss (RFLUSS)	Restored either S-Band or X-Band (or both) downlinks subsequent to a failure of either an exciter or transmitter.
Power Check (PWRCHK)	Configured the spacecraft to a safe power state in the event of an undervoltage condition, a main-to-standby inverter switch, or a CCS tolerance detector trip.
IRIS Power (IRSPWR)	Selected the infrared interferometer spectrometer and radiometer subsystem (IRIS) standby redundant heater unit if the prime unit failed.
AACS Power Code Processing (AAC SIN)	Allowed the CCS to respond to power codes from the AACS. This was a "hand shake" interface that allowed the CCS to monitor the health of the AACS computer and respond to anomalous conditions. Was also used by the AACS to command redundant or peripheral hardware during normal operation.
CCS Error (ERROR)	Responded to anomalous CCS hardware and software conditions and placed the CCS in a known, quiescent state awaiting ground action.
Tandem and Turn Support (TRNSUP)	Verified the integrity of the CCS and AACS prior to critical functions including turns and trajectory correction maneuvers.
Heartbeat Generator Self Test Control (HEARTBEAT)	Provided a heartbeat check of the AACS-CCS and communication link. Also acted as a performance check on the AACS processors.
Omen Power Code (OMEN)	Issued a power code to CCS indicating detection of a serious fault in AACS. Resulted in saving subsequent power codes for ground analysis. Also inhibited certain functions such as trajectory correction maneuvers.
Celestial-Sensor Fault Detection/ Protection	Monitored operations of the sun sensor and Canopus star tracker. On detection of error, triggered the reacquisition of celestial references and swapped AACS redundant elements.

TABLE III-6 (Sheet 2 of 2)

Voyager Spacecraft Fault Management Algorithms

NAME	FUNCTION
Power Supply Fail	Checked power monitors in the AACS and initiated a swap of redundant elements if a fault was detected.
Memory Refresh Fail	Monitored the plated wire memory cell refresh performed by the AACS processor. Initiated a processor swap if the refresh process failed.
Trajectory Correction and Attitude Propulsion Unit Failure (TCAPUF)	Performed thruster pulse rate and spacecraft angular position error checks to detect failures in thrusters. Switched redundant thrusters or AACS interface or processor units if a fault was suspected.
DRIRU Fault Protection	Monitored gyro (DRIRU) warmup and compared outputs during operation to determine faults. Swapped redundant gyro and AACS elements if a gyro fault was detected.
Scan Platform Slew Fault Detection/Protection	Software timer in AACS to detect slews which exceeded time allowed and prevented actuators from driving against stops. CCS response depended on frequency of fault occurrence.
Command Parity Fail	Performed a parity check in AACS on commands from CCS. If a parity error was detected, transmitted a power code to CCS and did not respond to the commands.
Command Sequence Fail	Protected against false commands to AACS/CCS interface and AACS processor and interface unit operation. Persistent absence of power code echoes resulted in swap of AACS elements.
Bad/No Echo Response	AACS power codes were echoed by CCS to test AACS/CCS interface and AACS processor and interface unit operation. Persistent absence of power code echoes resulted in swap of AACS elements.
Turn Complete and TCM Turn Abort	AACS aborted any turn if the sum of pitch, yaw and roll errors exceeded 6 degrees. Also rejected any turn commands if a turn was in progress.
Self Test	AACS conducted a self-test of the processor on request from CCS. If unsuccessful, a processor swap was ordered.
Catastrophe Handler/Processor Faults	AACS initiated interface unit and processor swaps if earlier switch to a redundant element had failed to correct a fault.

2.2 AUTONOMOUS NAVIGATION

An autonomous navigation subsystem should be capable of performing on board the spacecraft all of the navigation functions currently executed on the ground. These functions include determining the actual orbit of the spacecraft (knowledge), predicting future positions and events, and performing trajectory correction maneuvers (control). The measurement systems are typically self-contained.

2.2.1 Functions Supported

In theory, Autonomous Navigation can support the following mission and spacecraft functions:

- Instrument Pointing
- Orbital Stationkeeping
- Avoidance Maneuvers
- Communications
- Attitude Control
- Data Annotation
- Orbital Event Prediction
- Anomaly Detection
- Relative Vehicle Control

These functions are described in more detail in the following paragraphs

2.2.1.1 Instrument Pointing. Using the knowledge of the spacecraft's location relative to the Earth and the location of celestial objects (Sun, Moon, stars), an Autonomous Navigation system can compute the direction to a specified object as a function of time. The system can also determine if an observation is feasible. This enables both high-level commanding capability and automatic sequencing.

2.2.1.2 Orbital Stationkeeping. Certain missions require that the spacecraft maintain a specified orbital location. Geosynchronous stationkeeping provides the clearest requirements, however, other missions such as those requiring a particular Sun phase angle have similar requirements. The knowledge and predictive capabilities of an autonomous navigation subsystem enables the computation of the propulsive maneuvers required to maintain the required trajectory. Sufficient information is available to allow for automatic execution of the maneuvers.

2.2.1.3 Avoidance Maneuvers. Avoidance maneuvers include the general class of maneuvers which are executed based on some command external to the Navigation subsystem. Given this external command, the Navigation subsystem can compute and execute the desired maneuvers. Knowledge of the maneuver allows the updating of the onboard orbit knowledge and continued execution of the navigation mission functions.

2.2.1.4 Communications. In addition to providing antenna pointing information, an Autonomous Navigation subsystem can also compute relative velocity data (Doppler shifts) in support of narrow bandwidth communications. In other situations, the Navigation subsystem can provide orbits to enable ground tracking station pointing and timing prediction.

2.2.1.5 Attitude Control. Precision Earth relative attitude control may be limited by the accuracy of existing Earth sensors. Significant increases in precision may be obtained by using stars to establish an inertial reference coupled with an onboard spacecraft ephemeris which relates the inertial pointing to Earth pointing.

2.2.1.6 Data Annotation. Onboard knowledge of the spacecraft trajectory at the times data were taken allows for location annotation on the data prior to transmission to Earth. For example, latitude and longitude grid lines may be automatically added to a weather picture prior to transmission. This capability can not only decrease ground costs and processing time, but also enables immediate use by the user. This application is being demonstrated by a Landsat-D experiment using the Global Positioning System (GPS).

2.2.1.7 Orbital Event Predictions. Knowledge of the spacecraft trajectory plus the availability of a trajectory propagator in an Autonomous Navigation subsystem enables the prediction of a wide range of orbit dependent events. Typical examples include solar and lunar occultations, station acquisition and loss times, and picture opportunities. Automatic time updating of onboard sequences could significantly improve mission data return.

2.2.1.8 Anomaly Detection. In addition to estimation and control of the spacecraft orbit, an Autonomous Navigation subsystem can estimate other parameters such as sensor biases and maneuver magnitudes and directions. By comparing the estimated values to the predicted values, independent detection of certain anomalies may be achieved. This capability contributes to the overall autonomy of the spacecraft operation.

2.2.1.9 Relative Vehicle Control. The control of the relative motion of two or more vehicles in close proximity requires the application of Autonomous Navigation subsystems since the characteristic times of the motion are small relative to the ground reaction time. This type of navigation and control is required for both independent vehicles and vehicles with non-rigid

connections. The navigation sensors for these applications will, in general, be different from those used for single vehicle applications and are not discussed in this document.

2.2.1.10 Functional Hierarchy. A functional hierarchy for stationkeeping service is shown in Figure III-8. Ground based tracking support and functions that might be allocated to propulsion and attitude control subsystems are included as well as those appropriate to navigation. The set of functions directly applicable to an Autonomous Navigation subsystem lie primarily under the "Direct/Control Orbital Position" function. These are:

- (1) Process navigation sensor measurements.
- (2) Determine spacecraft orbital parameters.
- (3) Propagate the spacecraft ephemeris.
- (4) Schedule and compute trajectory control maneuvers.
- (5) Generate maneuver commands.
- (6) Verify navigation performance.

An additional important function not covered is to provide executive control of the overall navigation subsystem.

Functions (1), (2), and (3) accomplish the ephemeris maintenance service. While these functions are highly interrelated, certain options exist for the implementation of each function.

The following sections provide a generalized description for an Autonomous Navigation subsystem, along with the system level requirements, and discuss in some detail the first five of the above functional categories. These latter sections present more detailed functional descriptions along with various implementation options. The sixth functional category, "Verify Navigation Performance", has received relatively little analysis to date and will be discussed at only the top level. Continuing development within the Autonomous Spacecraft Program should provide additional data in this area. The final section presents a brief description of the Autonomous Navigation efforts to date along with a bibliography of reference papers.

2.2.2* Autonomous Navigation Subsystem Description

Figure III-9 presents a block diagram for a generic Autonomous Navigation subsystem. Depending upon the application not all of the elements may be required. There are also several methods for interconnecting the various elements of the system. For example, the measurement data could be routed through the Executive Controller rather than directly into the navigation subsystem. The components are:

*by J. B. Jones

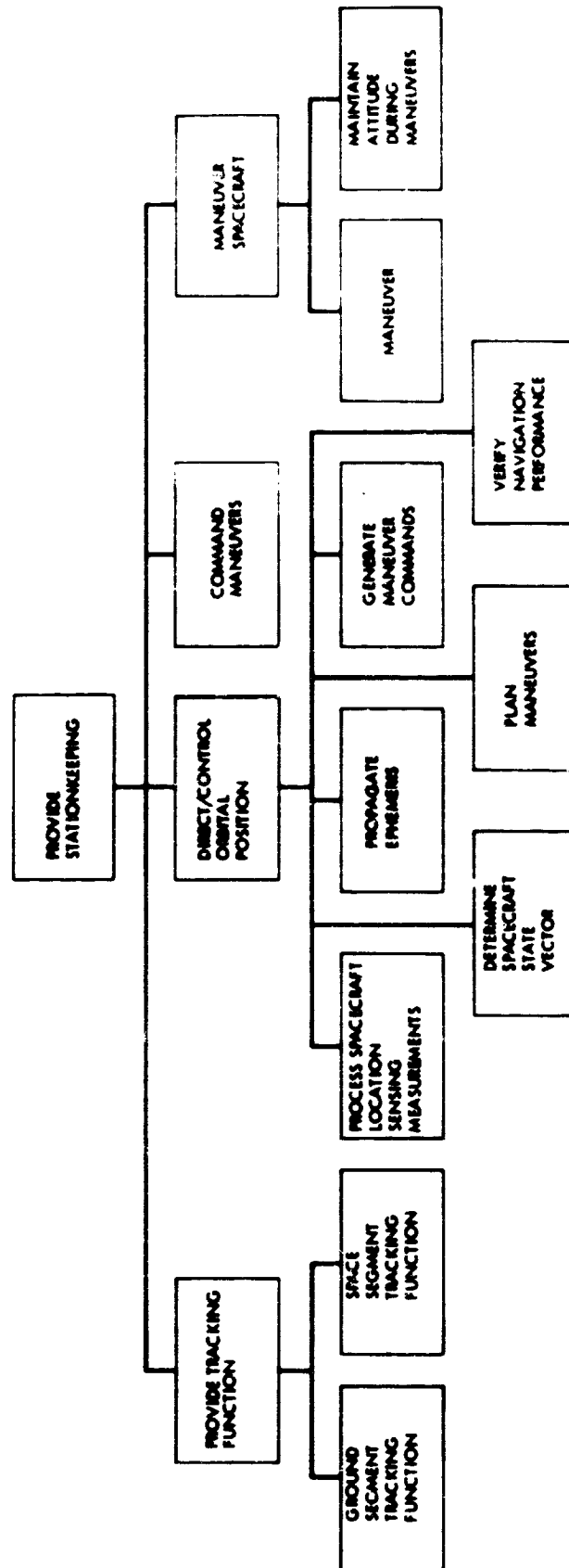


Figure III-8. Stationkeeping Service Functional Hierarchy

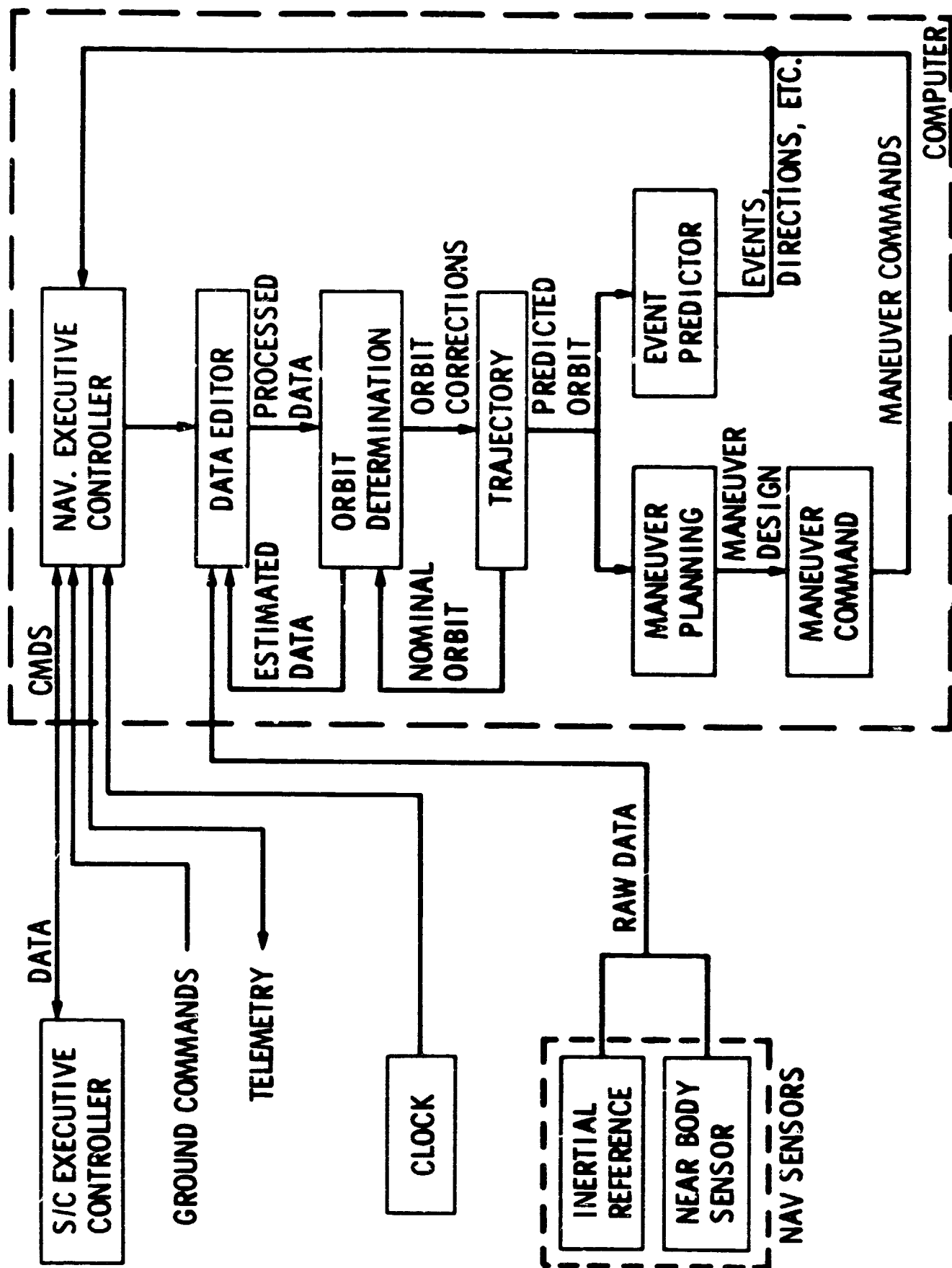


Figure III-9. Autonomous Navigation Generic Subsystem Block Diagram

2.2.2.1 Navigation Measurement Sensors. These instruments obtain the data required for navigation. They may operate continuously or upon command and they might be shared with the Attitude Control subsystem. Typical sensors include Horizon Scanners, Sun and Star sensors, Charge-Coupled Device (CCD) cameras, radars, and radio signals. The sensor section (2.2.4) presents more detailed data. The selection of the appropriate sensors depends upon the mission, the spacecraft design, and the accuracy requirements.

2.2.2.2 Data Editor. Prior to use in the orbit determination process, the sensor data must be edited to eliminate erroneous points and possibly calibrated to account for sensor characteristics. In addition to improving the estimation process, data from the editor supports detection and isolation of failures within the navigation system.

2.2.2.3 Orbit Determination. This is the heart of the navigation subsystem where the incoming data is used to improve the knowledge of the spacecraft trajectory. A wide range of algorithms based on estimation theory is available and the selection depends upon the particular application. Development of fail-safe hands-off algorithms may be a significant challenge.

2.2.2.4 Trajectory. The trajectory propagator services both the orbit estimator and maneuver planning segments in addition to producing unique output data. Commonality insures consistency between the various segments. Again, a wide range of techniques is available, and selection depends upon a detailed analysis of the mission characteristics and accuracy requirements. The trajectory segment produces output which includes the predicted spacecraft flight path, predicted orbital events, data annotations, and also includes the solar and lunar ephemerides.

2.2.2.5 Maneuver Planning. Within this block it is first determined if a maneuver is required and, if so, the time, direction, and magnitude of the maneuver is computed. Under some conditions a single orbit correction may be accomplished with more than one maneuver. Any other computations related to translation maneuvers are also computed in this block. The output is the "desired" maneuver(s) and may account for only a portion of the constraints. The detailed algorithms are both mission and spacecraft dependent.

2.2.2.6 Maneuver Commands. In this segment the previously generated maneuvers are translated into specific maneuver commands which may be executed by other spacecraft subsystems. A separate segment is defined since this segment is completely spacecraft peculiar. The functions assumed by this segment depend upon the overall autonomous architecture.

2.2.2.7 Clock. Figure III-9 assumes that the clock is external to the Navigation subsystem, however, for some applications it may reside within the Navigation subsystem. In any case the Navigation subsystem places strong requirements on the clock in terms of both accuracy and continuous timekeeping. Accuracy requirements range from 10^{-8} parts per day up to 10^{-11} and beyond. Many applications do not require highly advanced clock technology.

2.2.2.8 Executive. The executive segment provides three functions: coordination of the internal operation, communications with the outside world, and inter-segment fault isolation and detection. Historically, executive programs have been the most difficult segments to design, implement and validate.

2.2.2.9 Interfaces. In addition to responding to ground commands and providing telemetry data, a Navigation subsystem will have interfaces with:

- (1) Attitude Control - pointing and maneuvers
- (2) Propulsion - maneuvers
- (3) Electrical - orbital events, pointing and maneuvers
- (4) Data - data annotation
- (5) Payload - state vectors

2.2.3* System Level Requirements and Considerations

A number of mission and system related features drive a navigation subsystem design. These include:

- (1) Mission Requirements
- (2) Spacecraft Design Requirements
- (3) Characteristics of Other Spacecraft Subsystems
- (4) Operational Environment
- (5) Ground System

*By J. B. Jones

2.2.3.1 Mission and System Requirements. The most important driving factors are the Mission Requirements. They will determine many of the functions of the navigation subsystem and the level of performance. For example, one mission may require only the generation of spacecraft ephemeris while another may require the inclusion of full maneuvering capability. The mission requirements will also lead to the selection of particular orbital configurations which in turn may place constraints on the sensors which may be utilized. Geosynchronous orbits impose different constraints than do low polar orbits.

2.2.3.2 Spacecraft Design Requirements. Since an autonomous navigation subsystem (s/s), by definition, operates entirely onboard a spacecraft it must be designed in conformance with the overall spacecraft design requirements. These design requirements will place requirements on both the nominal and fault-tolerant operation of the Navigation subsystem. System requirements will dictate such factors as single point failure requirements, power considerations, and the fault detection/correction philosophy. The interfaces between the Navigation subsystem and the other subsystems is typically controlled at the system level in order to assure consistency and compatibility. The requirements for both subsystem level and system level test and validation are also specified by the system requirements.

2.2.3.3 Subsystem Characteristics. The third area which impacts the design of the Navigation subsystem is the characteristics of the other spacecraft subsystems. The primary interactions occur with the Attitude Control, Propulsion, TT&C, Power, and, possibly, Structure.

Attitude Control interfaces may arise in a number of ways. First the sensors may be shared. In general, the requirement placed on the sensors by the two subsystems will be different. Further, the protocol for sensor fault detection and correction must be clearly established and understood in order to preclude "deadly embrace" conditions in which two or more subsystems may be each awaiting a response from the other/others. Secondly, the navigation measurements may be sensitive to spacecraft attitude. And finally, certain attitude control systems may use the orbital ephemeris data to achieve accurate pointing. In this latter case, not only are requirements placed on navigation accuracy, but the operating characteristics must be consistent.

Propulsion/navigation interactions occur whenever propulsive maneuvers are conducted under autonomous control. In many cases, the Navigation subsystem generates the maneuver commands which are then executed by the Propulsion subsystem. In addition, the Navigation subsystem needs to have knowledge of any maneuvers which have been accomplished. For certain high capability Navigation subsystems, it is also possible for the Navigation subsystem to support Propulsion subsystem failure detection.

The TT&C and power interfaces are the usual subsystem interfaces. Structures interfaces may arise due to sensor field-of-view requirements.

In addition to these interfaces, which will occur on most spacecraft, some spacecraft may carry payload instruments or sensors which can supply navigation data. In these cases, the navigation must be carefully designed in order to interface efficiently and not degrade the primary payload functions.

2.2.3.4 Operational Environment. Operational Environment refers to both the orbit of the spacecraft and to the attitude control characteristics. Relative to the orbit, sensors which are appropriate for low altitude spacecraft may not yield a sufficient accuracy for high altitude spacecraft. Elliptical orbits impose special requirements on both sensors and algorithms. Functions such as the trajectory function are sensitive to the orbit. The trajectory modeling complexity varies with both the orbit and accuracy requirements.

The Navigation subsystem may be affected by the attitude control characteristics. Obtaining the required sensor data will be much easier on a very stable spacecraft as opposed to one with a dynamic attitude. Spinning spacecraft will require many different considerations.

2.2.3.5 Ground System. As with any other subsystem, the Navigation s/s must interface efficiently with the ground control system. In order to provide satisfactory ground control, data must be transmitted on both the inputs and outputs of the Navigation subsystem. Sufficient data should be transmitted to enable ground validation of the space system performance through reconstruction of the space subsystem function.

2.2.4* Measurement Sensors for Autonomous Navigation

The navigation computer must be supplied data from three classes of sensors: inertial references, near-body sensors, and clocks. Fig. III-10 illustrates the situation and lists some possible ways of accomplishing the required sensing. Certain generalizations can be made based on the current state of development of the various sensors.

2.2.4.1 Inertial Sensors. The most common attitude reference is the celestial sensor. This determines directions in inertial space by sighting on celestial objects sufficiently distant from the spacecraft that they may be regarded as fixed. Star sensors and sun sensors both fall in this class. Strictly speaking, no celestial object is fixed since all stars show the phenomenon of aberration as a result of this motion of the observing telescope, but this has a small affect on orbit determination, amounting to +20 arc-sec for orbital motion of the earth and to + 5 arc-sec for a low satellite. Likewise, with a sufficiently sensitive sun sensor, the apparent motion of the sun against the background of fixed stars as a result of the motion of the spacecraft could be measured, and the sun would become a near object rather than a celestial object. Although this motion is larger than the aberration for typical earth-orbiting spacecraft, it is still too small

*By E. F. Tubbs

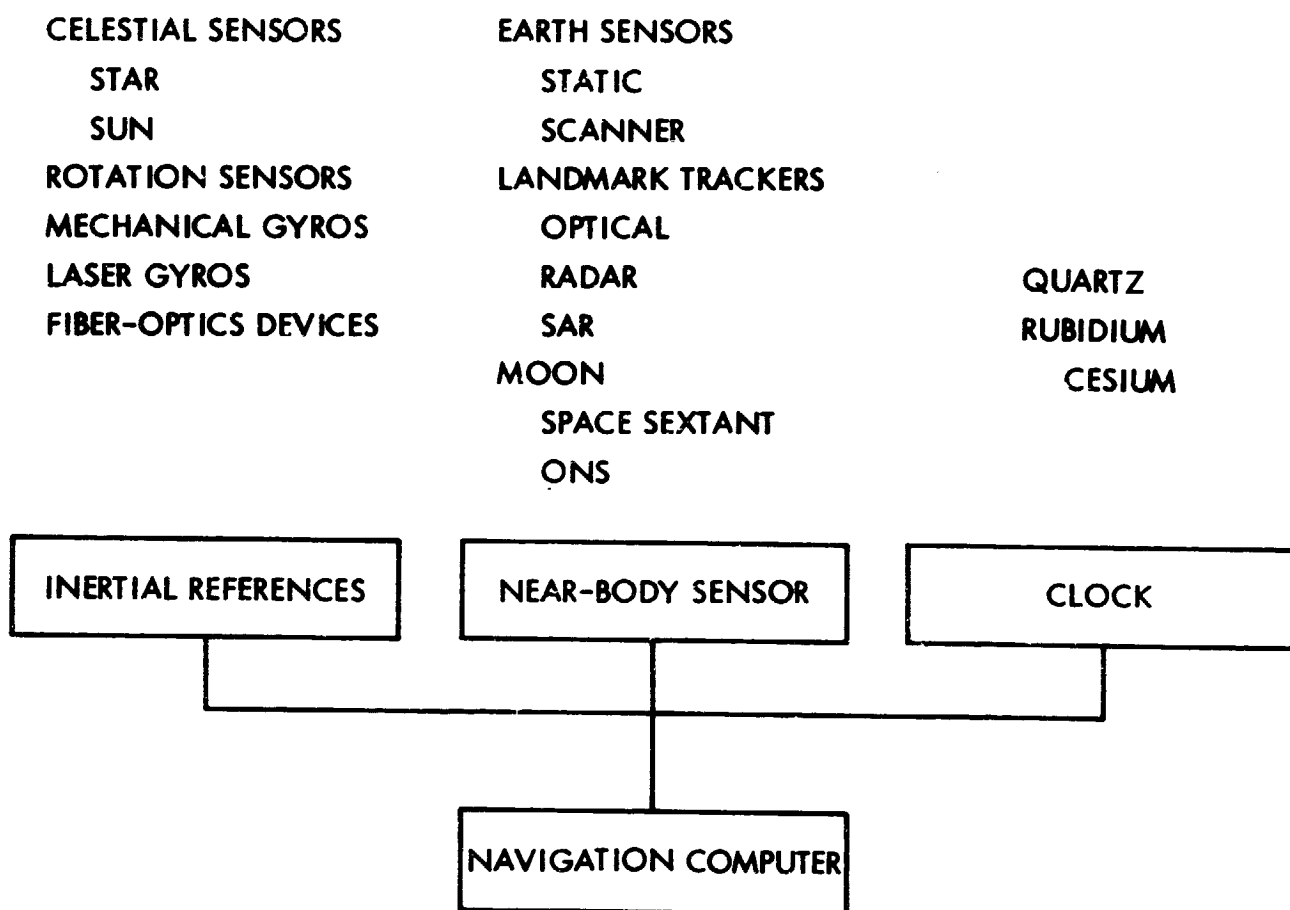


Figure III-10. Sensors for Autonomous Navigation

to limit most practical navigation. Both stellar aberration and solar parallax can be corrected for by computation if necessary, and this places the sun and stars firmly in the category of celestial objects suitable for attitude reference.

One question remains with respect to celestial reference: How good a correction can be made for the effect of proper motion? The Fourth Fundamental Catalogue FK4, which was prepared by W. Fricke, A. Kopff and coworkers in 1963, contains 1535 fundamental stars brighter than a visual magnitude of 7.5 and distributed over the sky with fair uniformity. There is also a supplement containing 1987 stars. This will be superseded by FK5 which is expected to contain about 3500 stars to a magnitude of 9.2. Although the uncertainty of the proper motions will vary from star to star, the average uncertainty is estimated to be about 2 milliarc-sec/year for stars in FK4. Therefore guide stars taken from FK4 can be corrected for proper motion with sufficient accuracy. There is a general correction to reduce the coordinate system of the Catalog to an inertial frame. Various determinations of these corrections have yielded different results, but the results are on the order of 5 milliarc-sec/year with uncertainties of 2-3 milliarc-sec/year. These corrections are only of importance in situations requiring extreme accuracy in the determination of spacecraft attitude such as a relativity experiment.¹

The other attitude references listed in Fig. III-10 are what might be called internal ones, in that after initialization they do not require the observation of external objects. Mechanical gyros are used in present-day spacecraft only at critical times such as orbit insertion as they do not have the life to allow continuous operation on long-duration missions. Although the wear-out mechanism is different, the same can be said of the laser gyro. At the present time the fiber-optic rotation-sensing devices offer great promise as a continuously operating internal reference, but are still in the laboratory.

The configuration used for celestial sensing will depend upon the spacecraft and the mission. Typical configurations are two or three strap-down star sensors or a star sensor and a sun sensor for three-axis stabilized spacecraft. With a spinning spacecraft the sensor takes the form of some type of scanner. One such is the V-slit scanner, which senses the time of passage of known stars across the two legs of the "V".

In some situations it may not be necessary to have complete inertial reference. If it required that only the in-plane motion of a satellite be controlled, it is sufficient to have a celestial sensor aligned in or near the orbital plane. Likewise if only the orientation of the orbital plane must be controlled, the most convenient arrangement is to have a sensor directed towards a "pole star" of the orbit.

The choice of strap-down star sensors is limited at the present time. The only one immediately available is the NASA Standard Star Tracker built by Ball Aerospace. It uses a magnetically focused and deflected image-

dissection tube. Its specifications are given in Table III-7. Various imaging star trackers using solid-state detectors are under development. The Air Force Multimission Attitude Determination and Autonomous Navigation (MADAN) program is developing a tracker based on a radiation hardened charge-coupled device (CCD) built by Hughes. NASA (Ames and JPL) has developed a prototype fine-guidance sensor for the Shuttle Infrared Telescope Facility (SIRTF) based on an RCA CCD which, although not strictly a star tracker, is close to a startracker configuration. The RCA CCD is not radiation-hardened. At this writing the spacecraft designer can consider only the NASA standard as an available item. Its accuracy of 10 arc-sec over a field of view is only achieved by a careful ground calibration which yields 190 calibration coefficients. Once this is done, it seems to perform satisfactorily. The Magsat mission operated with redundant attitude-determination devices, including two star trackers. A comparison between the direction as determined by one of the trackers and the directions determined by the

TABLE III-7
NASA Standard Star-Tracker Functional Characteristics

<u>NASA Standard Parameter</u>	<u>Value</u>
Photo-Sensor Type	Image dissector, ITT F4012 KP
Spectral Response	S-20
Lens	70mm, f/1.2
Window	Fused quartz, 0.3 in. thick
Field of View	80 x 80
Sensitivity	5.7 visual magnitude, Class 60V
Vehicle Rate	0.30/sec
Search Mode	Raster Scan
Acquisition Time	10 sec
Track Model	Unidirectional cross scan
Output Data Rate	10 updates/sec each axis
Track-Scan Period	100 msec total for both axes
Noise-Equivalent Angle	16 arc sec maximum
Total Accuracy	10 arc sec (1) over total field
Analog Outputs	Star magnitude, instrument temp.
Digital Outputs	Two 16-bit words giving star position (12 bits for each axis) and status
Size	6.5 x 7 x 12 in basic tracker
Weight	17 lbs. only

second tracker in combination with the sun sensor shows a variation in the difference of 5 arc-sec rms. Drifts in alignment correlated with temperature were observed during the Magsat mission. Part way through the mission a heater on one of the trackers failed. This caused a shift in direction of 6 arc-sec.²

Although CCD trackers are not available as flight-qualified devices at the present time, their performance can be projected from the results of laboratory and field tests. The advantages of the CCD are that the geometry is fixed by the physical structure of the detector and they are much less sensitive to magnetic fields. The uncertainty in star location is typically one part in 10^4 of the field of view. A CCD tracker with the same 80° field as the NASA standard would have an uncertainty of 3 arc-sec without calibration. The sensitivity depends upon the collection efficiency of the optics. A reasonable estimate is useful operation at a visual magnitude of 6 or 7.³

The situation with respect to sun sensors is significantly different. The sun sensor of choice for a 3-axis stabilized spacecraft is the NASA standard made by Adcole Aerospace Products. It covers a field of view of 640×640 so that a total of six are required to cover a full circle. Each sensor head contains coarse and fine reticles for measurement of sun angle in two orthogonal directions. The specified accuracy of the sun sensor is +60 arc-sec within a 300° cone angle. The repeatability is specified as +30 arc-sec. The experience with Magsat indicates that with ground calibration and a significant amount of onboard calculation a resolution of 2 arc-sec and an accuracy of 12 arc-sec rms could be obtained. The experience with Magsat also showed a fixed discrepancy of 55 arc-sec between the ground calibration and the flight measurements in one axis only. The origin of this was not determined, but it indicates that provision must be made for in-flight calibration of the sensor pointing directions when high accuracy is required. In addition to Adcole sun sensors a wide variety of sensors has been built by TRW, Honeywell, Lockheed, Ball Brothers, Hughes, and Bendix.

2.2.4.2 Near Body Sensors. Of the various near-body sensors, the earth sensors are the most highly developed with substantial flight experience with both static and scanning devices. The available earth sensors detect the horizon in either the 14-16 micrometers CO_2 band or in the 22-40 micrometers H_2O band. Sensors operating in the CO_2 band typically use germanium optics while those in the H_2O band use silicon. Detection is done with thermal detectors: thermocouples, thermopiles, or bolometers. Sensors are manufactured by Barnes Engineering, Ithaco, TRW and Quantic Industries.

The Quantic Industries Model 5100 serves as an example of a static instrument. It is a radiation-hardened instrument designed to operate at geosynchronous altitudes. It senses pitch and roll using radiation-balance on eight thermocouples and operates in a band from 22 to 33 micrometers. In addition to the horizon-sensing optical system it has a separate system to sense the presence of the sun in the field of view of the detectors. This

sun-presence system automatically shuts down any of the thermal detectors directly affected by solar radiation. The sensor has a linear range of ± 2 degrees and a saturated signal range of $\pm 17^\circ$. The uncertainties at geosynchronous altitude are given in the following table.

Source of Uncertainty		3 sigma
Electronic noise		0.002 deg.
Electronic offset		0.005
Long-term drift in detectors		0.015
Changes in housing temperatures		0.008
	RSS	0.018
Alignment and calibration		0.030
	RSS	0.035
Horizon variations		0.022
	RSS	0.041

The line of horizon scanners made by Ithaco serves as an example of the scanning class of sensor. It uses germanium optics and an immersed bolometer as a detector. A wedge which gives a deflection is rotated in front of the lens to give a conical scan. A typical configuration places a pair of sensors on the spacecraft with the axis of the detectors aligned with the + and - directions of the pitch axis. Each sensor generates a square wave as the scan passes on to the disc of the earth. The phase of the wave is a measure of pitch attitude, while the duty cycle is a measure of changes in roll. "Peaking circuits" are required to make the rise and fall in the square waves as independent as possible of variations in horizon temperature. These sensors may be used from 150 km to above geosynchronous altitudes.

It is necessary to make correction for variations in the horizon location with season and latitude as well as for the flattening of the earth. Once this has been done there remain random effects given in the following table:

Source of Uncertainty	3 sigma at 1000 km	
	Roll	Pitch
Horizon random-radiance effects	0.022 deg	0.028 deg
Electronics drift	0.015	0.015
Optical alignment	0.02	0.02
Noise (one sample)	0.034	0.047
(four samples average)	0.017	0.024
Quantization error	0.005	0.005
RSS (one sample)	0.048	0.060
(four sample average)	0.037	0.046

Landmark tracking has been the subject of several studies over the past 10 years. Some of the possibilities which have been or are being investigated are listed in Fig. III-10. They are the optical tracking

of landmarks, the tracking of radar transmitters and the use of synthetic-aperture radar (SAR). The optical-tracking investigations were concerned with unknown landmarks, while the radar studies have been concerned with known landmarks.

Both TRW⁴ and Honeywell⁵ made studies of the optical tracking of unknown landmarks. The TRW work resulted in an engineering model of a gimbaled tracker using an image dissector tube as the detector. This system was carried through the laboratory-test phase. The performance was one sample every 10 sec with a gimbal bias of 4 arc-sec and a random error of similar size. The Honeywell system utilized two body-fixed silicon-matrix photo detectors to measure the landmark motion. This system carried the sensor assembly through the critical component development phase. Performance on the order of 1 km for low altitude orbits was projected.

The tracking of ground-based radars at known locations has been studied by IBM.⁶ The system used interferometric techniques to provide accurate tracking of radar landmarks. The center frequency is 3 GHz which provides all-weather operation and limits ionosphere refraction to 4 arc-sec. This program proceeded through the critical-component development phase, and predicted accuracy levels ranged from 68 meters (1 sigma) for the Molniya 12 hr. orbit to 900 meters for a synchronous orbit. In low earth orbit the system can also provide attitude information to ± 1 arc-min without the use of celestial sensors.

A possible approach to known landmark tracking is synthetic aperture radar (SAR). Such a system would use SAR images in combination with a star or sun sensors to provide navigation data. Beginning with an initial orbit estimate, the system first selects a landmark and commands a navigation image. At the appropriate time the SAR takes the image which is then processed. In parallel, the navigation system selects the proper map from its file and performs the necessary rotation and scaling. The updated map is then correlated with the SAR image and the navigation data is extracted. Landmark locations should be easily determined to 100 meters, and with additional work, including possibly inflight calibration, determined to 10 meters or less. Thus for reasonable SAR range accuracies, navigation accuracies to 200 meters or less are possible. A fine-tuned system might produce accuracies in the 10-20 meter region for orbital altitudes in the range of 250 to 1000 km. No work is known to be under way in this area at this time.

The remaining near-body tracking possibility is the moon. There are two sensors suitable for moon tracking under development at the present time. The first of these is the Space Sextant being developed by Martin-Marietta for Air Force Space Division.⁷ The principle of operation is the measurement of the angle between the limb of the moon and a known star by measuring the angle between two telescopes, one tracking each object. It can also measure attitude by measuring angles from known stars to reference surfaces in the sensor. The mechanical arrangement of the sensor places the two telescopes in a common plane and orients this plane with a gimbal assembly to allow the desired measurement. A spinning wheel concentric with the telescopes injects a collimated timing light into each telescope. The timing of these light pulses measures the angle between the telescopes. There is a wide field of view used for acquisition and a narrow, high-resolution field for the fine measurements. The resolution of the angular measurements is about 0.15 arc-sec and the accuracy is of the order of one arc-second.

The limb-tracking telescope is driven by the servo system to that part of the limb normal to the star direction. The computation must correct for the radius of the moon and for significant variations in the limb as a result of topographic variation. The result is a measurement of the angle between the center of mass of the moon and the star. One measurement locates the spacecraft on the surface of a cone with its apex at the moon and axis in the direction of the star. A second measurement to a different star establishes a second cone. The intersection of the two cones establishes a line of possible positions to the spacecraft. A sequence of lines of position and their times enables the spacecraft position to be established. With the aid of an onboard estimation filter it is projected that Space Sextant can yield positioning to 250 meters, 1 sigma, and attitude to 0.6 arc-sec, 1 sigma, in any orbit in the earth-moon system.

The recently developed technology in charge-coupled device (CCD) imaging detectors, which was discussed above in connection with star trackers, also has application to near-body sensing. An example of work in this area is the Optical Navigation Sensor under development at JPL for NASA.⁸ This is an astronomical optical-navigation instrument employing single-frame, slow-scan television imaging. It uses a commercial CCD area-array detector in combination with a microprocessor which extracts near-body and background-star centers from each frame. Its application to lunar tracking is facilitated by the well-defined lunar limb.

For limb finding, each line is scanned for groups of pixels which both exceed a specified threshold and exhibit a rapid change of intensity. Groups of pixels satisfying these criteria are stored for later processing. Since these pixels are the only ones stored, most of each scan line is discarded, and a data compression of 10 to 1 is achieved without loss of metric accuracy. After the limb-finding operation, a small band of pixels outlining the limb and containing some terminator points will have been stored. The processing then removes obviously bad points and terminator points and makes a preliminary estimate of the center. Limb points are then computed for each line by finding the point where the directional derivative is greatest in the direction of the estimated center. These limb points are then fitted with an ellipse using linear mean square error estimation. The lunar center is then defined as the geometrical center of this ellipse.

The centerfinding can be done to an accuracy of ± 3 micrometers on the CCD. For a 150 mm focal length lens this translates into ± 4 arc-sec and is proportionally smaller for lenses of longer focal length. A breadboard of the Optical Navigation Sensor is under construction, and field tests of the instruments on lunar centerfinding are planned.

2.2.4.3* Clocks. As previously noted in Figure III-10, there are three types of oscillators currently available for use in an onboard clock; quartz, rubidium and cesium. Of the three, the quartz technology is the best developed, with a long history of quartz oscillators culminating in a currently available and flight-proven high stability oscillator. The rubidium and cesium technology is newer, with the first generation of flight oscillators being used in the Global Positioning System Navstar satellites. These latter oscillators are expected to be fully flight-qualified in the near future.

*By J. B. Jones

Considering only the navigation requirements, the choice depends upon both the required navigation accuracy and the required period of autonomy. For example, a mission with a 10-meter navigation accuracy requirement over a 6 month autonomy period will require a much more accurate clock than a mission with a 1 km accuracy requirement over a 2 week period.

In order to quantify the relationship between oscillator accuracy and navigation accuracy, we may use the following first-order expression:

$$P = V_C \delta t$$

$$= V_C \delta s + \delta r (\delta f)t + 1/2 (\delta d)t^2$$

where P is the position error and V_C the orbital velocity. The timing error, δt is separated into the following types of errors:

δs = the error in initially setting the clock
 δr = the error in reading the clock time
 δf = the error in the knowledge of the oscillator frequency
 δd = the error in the knowledge of the frequency drift

Table III-8 presents typical values for the various error terms for the three types of oscillators. These ranges are fairly conservative and improvement may be possible with an optimized system. Assuming that all of the clock errors are independent, Figure III-11 presents the range of navigation position uncertainty due to clock error for the upper and lower bounds given in the Table and autonomy periods up to 180 days. A low altitude orbit (250 km) is assumed. For higher altitude orbits the error reduces by the ratio of orbital velocities. At geosynchronous altitudes, the errors are reduced by a factor of 2.7. As may be seen the lower bound of the errors is determined by the set and read errors. The frequency knowledge error and frequency drift errors for the cesium oscillator do not significantly contribute to the Navigation error.

The question of clock autonomy has not been considered in detail, however, preliminary considerations indicate that it would be desirable to operate three clocks simultaneously in order to detect and isolate internal clock failures. Further, to protect the clocks from external power fluctuations, the clocks should be provided with backup power supplies, i.e., batteries.

REFERENCES FOR TOPIC 2.2.4

1. Fairbank, W. M., et al., "Report on a Program to Develop a Gyro Test of Ground Relativity and Associated Control Technology," Stanford University, June 1980.
2. Foutain, G. H., et al., "The Magsat Attitude Determination System," John Hopkins A.P.L. Technical Digest, Vol. 1, No. 3 (1980).

Table III-8. Clock Error Characteristics

ERROR	OSCILLATOR TYPE		
	QUARTZ	RUBIDIUM	CESIUM
δ_s - INITIAL ERROR IN CLOCK SETTING	10^{-4} TO 10^{-5} SEC		
δ_r - ERROR IN READING CLOCK TIME	10^{-3} TO 10^{-4} SEC		
δ_f - FREQUENCY ERROR, SEC/SEC	10^{-10} TO 10^{-11}	10^{-11} TO 10^{-12}	10^{-12} TO 10^{-13}
δ_d - FREQUENCY DRIFT, PARTS/DAY	10^{-10} TO 10^{-11}	10^{-11} TO 10^{-12}	$<10^{-13}$

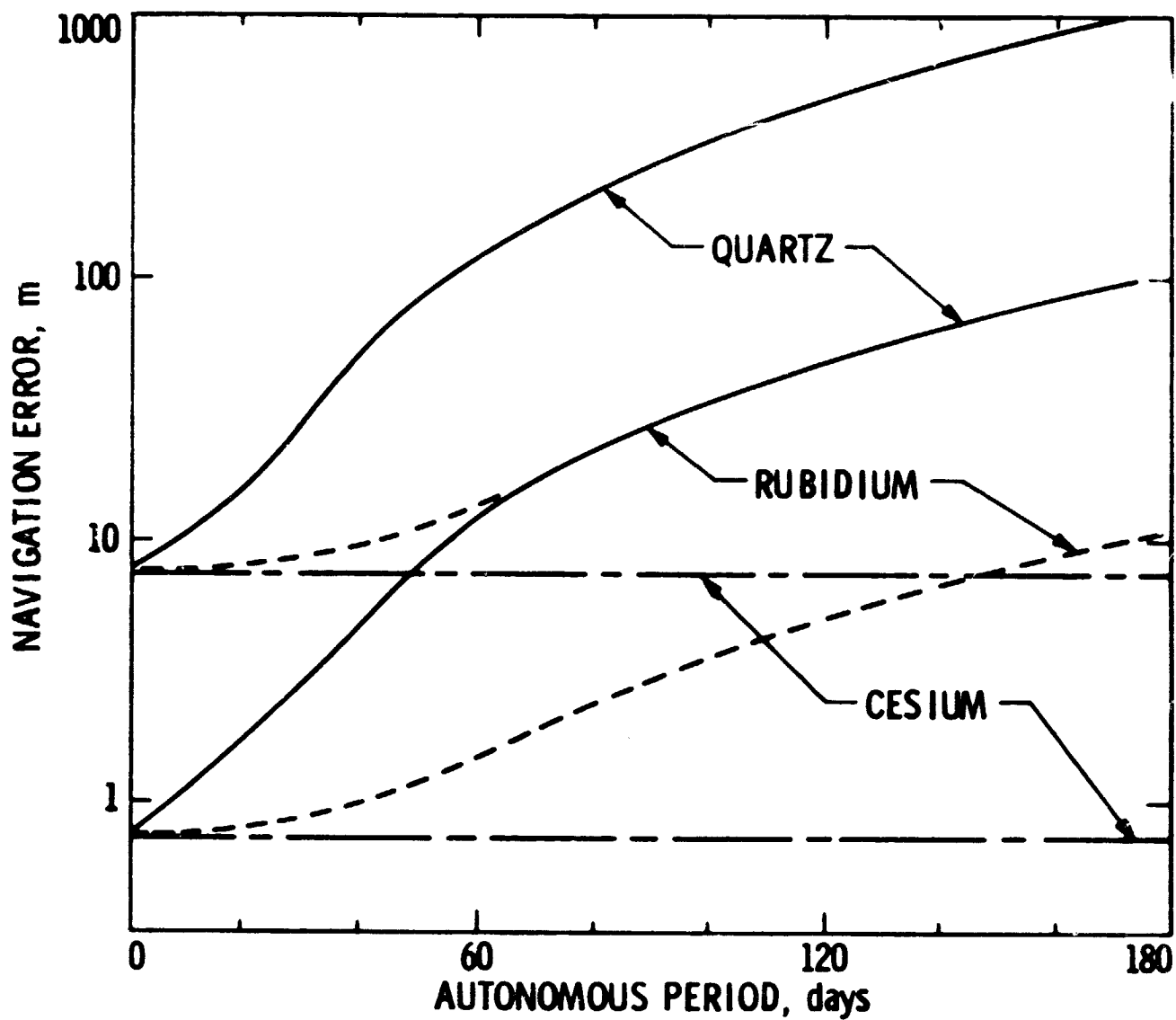


Figure III-11. Navigation Position Uncertainties Due to Clock Error

3. Salomon, P. M., et al., "Shuttle Pointing of Electro-Optical Experiments," Proc. SPIE 265. 203 (1981).
4. Williams, J. W., "Autonomous Navigation Technology - Phase 1, TRW, SAMSO TR No. 74-244, August 1974.
5. Paulsen, D. C., "Autonomous Navigation Technology - Phase 1A Study," Honeywell, SAMSO TR No. 74-221, March 1975.
6. Fritz, R. L., et al., "Autonomous Navigation Technology (ANT) System Phase 1 Final Report", IBM SAMSO TR No. 74-201, September 1974.
7. Booker, R. A., "Space Sextant Autonomous Navigation and Attitude-Reference System's Flight Hardware Development and Accuracy Demonstration", Rocky Mountain Guidance and Control Conference, March 1978, AAS 78-124.
8. Eisenman, A. R. and Armstrong, K. W., "CCU Sensors for Interplanetary Spacecraft Navigation and Pointing Control," Rocky Mountain Guidance and Control Conference, January 1981, AA 81-U21.

2.2.5* Process Measurements (Data Conditioning)

Before measurements are passed to the orbit determination subsystem for state estimation it is essential to screen the data to eliminate occasional "blunder points" and, for most applications, to perform other editing, calibration, and performance-monitoring functions as well. This combination of functions is sometimes called data conditioning.

2.2.5.1 Requirements on Data Conditioning. Requirements on data conditioning are strongly mission and spacecraft dependent, but in general the following functional requirements are applicable:

- (1) Prevent obviously bad measurements from reaching the state estimation function.
- (2) Eliminate excessively redundant state estimation computations by appropriate data selection and/or compression.
- (3) Apply calibrations to raw measurements (if available and not applied elsewhere) to enhance accuracy.
- (4) Provide audit data on orbit determination and sensor performance as the basis for anomaly detection.

*By J. P. McDanell and K. D. Mease

2.2.5.2 Identification of Bad Data. Bad measurements can be identified by testing pre-fit observation residuals for consistency with a predicted residual variance. Computation of the predicted variance is straightforward using the state covariance (or equivalent factors) and observation partial derivatives from the orbit determination process and the observation noise variance. If the ratio of the squared residual to its predicted variance is greater than a specified threshold value, the point is deleted. Since the main purpose of this test is to eliminate occasional blunder points, the threshold should not be set too low. Typical values might be in the range of 9 to 25 (where a value of 9 corresponds to a 3 sigma test and 25 is a 5 sigma test).

Another method of identifying bad data that should not be overlooked (when applicable) is a direct consistency check between redundant measurements. The comparison may be between redundant sensors of the same type or between sensors of different types measuring related quantities. If the measurements do not agree within allowable limits, additional tests (e.g., on residuals) will be required to determine which is correct.

2.2.5.3 Performance Monitoring. In conjunction with the screening of measurements some useful performance monitoring can take place. The average residual value, the residual sum-of-squares, and the number of points failing the screening test in a particular time period can readily be monitored and used to flag potential anomalies so that appropriate corrective action may be initiated at the system level. In general, anomalous residual behavior (e.g., a large bias on several successive measurements) may be due either to sensor malfunction or divergence of the orbit determination process. The particular diagnostic criteria and corrective actions to be taken are application dependent.

2.2.5.4 Data Compression. When highly redundant measurements are available, it may be wasteful of computer resources to perform a complete state estimate update with each measurement. On the other hand it is desirable to take advantage of the additional reliability inherent in redundancy. Thus some form of data compression or averaging may be appropriate. Usually this involves a simple arithmetic averaging of several independent measurements. The state estimator then treats this average as a single measurement with variance inversely proportional to the number of original measurements.

2.2.6* Determine Spacecraft State Vector

The orbit determination (OD) function maintains and continually corrects an estimate of the spacecraft state vector (position and velocity of orbital elements) based on information inherent in the measurements provided by the sensors. The OD process is primarily a data-fitting process consisting of the following steps:

- (1) Predicted observable values are computed at times which correspond to the acquisition times of actual realized observations (measurements).

- (2) Partial derivatives are computed, which relate how changes in the predicted observables at each time are affected by changes in the spacecraft state and (in some cases) other parameters of the spacecraft motion.
- (3) The difference between each observation value and its predicted value is computed.
- (4) These differences, or data residuals, are reduced using the associated partial derivative matrix, to an estimate of the spacecraft state, and if desired, the associated trajectory model parameters. These adjusted parameters then yield the trajectory (or orbit) which most closely predicts the observed data; i.e., drives the sum of the squares of the residuals to a minimum value.

2.2.6.1 Requirements on OD Function. Design and implementation of the OD function may be characterized as a tradeoff of computational requirements against accuracy and reliability. The computational requirements for OD are considerable, even in relatively simple systems, and typically are a major driver on the capacity of the onboard computer. On the other hand, the performance of the navigation subsystem, while ultimately limited by the sensor accuracy, may be artificially limited by the choice of models and algorithms in the OD function.

Clearly the specific requirements on the OD function are mission dependent, but the following general statements can be made:

- (1) The computational requirements of the OD function must be consistent with the numerical precision and speed of the onboard computer.
- (2) The completeness and complexity of the modeling must be consistent with the sensor accuracy.
- (3) The data fitting process must be tolerant of unmodeled dynamic effects, systematic measurements errors, nonlinearities, and all other potential error sources; i.e., the response to unexpected errors must be modest degradation of accuracy rather than divergence.

2.2.6.2 Estimation Algorithm Selection. At the heart of the data fitting process is an estimation algorithm. The literature abounds with such algorithms, and while many candidate algorithms for onboard OD applications can be shown to be mathematically equivalent under idealized assumptions, the differences in real-world performance in a computer of limited word

length, limited core, and limited speed can be significant. This section will define the major classifications of algorithms, describe their relative strengths and weaknesses, and provide some guidelines for selecting an algorithm for onboard OD.

Algorithms typically used for OD are built around a linear estimator. The estimator processes measurements to yield additive corrections to an a priori spacecraft state vector about which the nonlinear equations of motion and equations relating the spacecraft state to the observables have been linearized. Algorithms can be classified according to the mode of measurement processing. "Batch" algorithms accumulate all the measurements to be employed in a given orbit determination and process them simultaneously. "Sequential" algorithms process the measurements one at a time, as they are acquired. Sequential algorithms are clearly attractive if storing the required number of measurements for OD exceeds the storage capacity of the onboard computer. However, if the actual trajectory is a nonlinear function of the a priori reference trajectory, then more than one iteration may be required for convergence. This entails relinearizing about the corrected trajectory from the previous iteration and then reprocessing the measurements. Thus, the measurements must be stored for either the batch or sequential algorithms. Modified sequential algorithms, referred to as "extended sequential" algorithms, seek to avoid storing the measurements despite the presence of nonlinearities. The corrections produced by the linear estimator after the processing of a measurement are used immediately to correct the state vector. Then, before the next measurement is processed, a relinearization about the corrected vector is performed. The extended sequential algorithms require an increased amount of computation for a single iteration, but, on the other hand, they offer a higher probability of converging in the first iteration.¹

The sequential and extended sequential algorithms have a computational advantage over the batch algorithm. An essential step in the batch processing of measurements is a matrix inversion. If the measurements are processed sequentially, the matrix inversion is reduced to a sequence of scalar divisions, provided the measurements are uncorrelated. Another advantage of the sequential and extended sequential algorithms is the relative ease and computational efficiency with which dynamic process noise can be included. More will be said about this in 2.2.6.3.

In the presence of nonlinearities, the convergence of the batch, or sequential algorithms which process all the measurements before relinearizing, is less sensitive to an occasional "bad" data point than that of the extended sequential algorithms. A further advantage of the batch algorithms is that they are computationally more efficient for a large number of measurements.

Estimation algorithms can also be classified as to whether the covariance matrix for the estimated states is used in the computations or whether some factor or factors of it are. The Kalman sequential algorithm² involves the covariance matrix; the Potter sequential algorithm³ involves a square-root of the covariance matrix; the Carlson-Cholesky algorithm⁴ involves a triangular square-root of the covariance matrix; and the Bierman algorithm⁵

involves a UDU^T factorization of the covariance matrix, where U is a unit upper triangular matrix and D is a diagonal matrix. Somewhat analogously, there is the standard weighted least-squares batch algorithm, and there is the square-root information filter (SRIF)⁵, an alternative batch algorithm.

The "square-root" algorithms are more robust in the presence of numerical errors due to finite precision arithmetic⁶. This is an important property for an algorithm which is to be implemented on an onboard computer of limited word length. A corollary is that the square-root algorithms can tolerate a greater dynamic range of variables without becoming numerically unstable. A number of modifications to the Kalman algorithm have been suggested for the purpose of rendering it more numerically stable. However, the more fundamental approach of reformulating the algorithm in terms of a factorized covariance matrix has proven to be more effective.⁶

The Bierman U-D algorithm has the numerical robustness of the square-root algorithms, yet does not require the computation of numerous square-roots and is, therefore, faster than the square-root algorithms. Moreover, if it is efficiently implemented, its core requirements and computational speed are comparable to the more popular but less reliable Kalman algorithm⁶. Thus, the Bierman algorithm has the most desirable combination of properties for most onboard OD applications.

2.2.6.3 Formulation of Models. Integral parts of the onboard OD system are the mathematical models which are used to generate the predicted observables. The predicted observables are computed as a function of the free parameters in the models. The estimation algorithm determines the values of these parameters such that the predicted observables agree with the actual data in a weighted least squares sense. The free parameters consist of the components of the spacecraft state, along with other parameters appearing in either the trajectory or observation models. Because of the limitations imposed by the onboard computer, it is desirable to keep the number of free parameters, and hence, the dimensionality of the estimation problem, as low as possible without degrading the OD below the required accuracy.

There are a number of choices for the components of the spacecraft state (e.g., positions and velocities in Cartesian coordinates, classical orbital elements, etc.). The particular choice of components, in turn, dictates the form of the trajectory and observation models. Thus, there is some flexibility in formulating the problem which the estimation algorithm will be required to solve; a prudent choice may be an important factor in maximizing the ultimate accuracy and reliability of the onboard OD system.

Orbital elements are often preferable for modeling a spacecraft in orbit about a body. However, one must be careful to choose a set of elements which will not become singular over the range of possible trajectories. Each set of elements will lead to corresponding equations of motion. It may

be possible that, for a particular set of elements, the corresponding equations of motion can be solved analytically. This would then allow the partial derivatives needed for OD to be derived analytically, rather than by integration of variational equations, which would reduce considerably the computational load of the OD function.

Another possibility is that, for a certain set of elements, one or more of the elements may remain constant for the particular trajectory (orbit) to be flown. If the value of a constant element is known to sufficient accuracy, then there is no need to estimate it, and the dimensionality of the estimation problem can be reduced. Certain components of the spacecraft state may also be eliminated as estimated parameters, even if they are not accurately known, if knowledge of their values is not required and they are not observable, i.e., they do not influence the measurements.

Another important question is how accurate should the trajectory model (equations of motion) be? Should one account for third body effects, solar radiation pressure, asymmetries in the gravitational potentials of the gravitating bodies, spacecraft gas jet leaks, etc.? The considerations are: 1) storage capacity of the onboard computer, 2) the number of additional free parameters that will have to be estimated, and 3) the impact on the accuracy and reliability of the OD if the above items are not accounted for.

Since it is often not possible to model the trajectory to an accuracy consistent with that of the measurements, techniques for model error compensation are necessary. The techniques involve the addition of stochastic acceleration terms to the equations of motion. The stochastic accelerations can be modeled as white noise or Gauss-Markov processes of first, second, or third order. Gauss-Markov processes have been particularly effective in compensating for unmodeled spatial variations in the gravitational potentials of the Earth and the Moon. The stochastic accelerations, also known as "process noise", are easily incorporated into the sequential and extended sequential estimation algorithms. Their inclusion into the batch algorithms is more complicated and entails an increased computational load.

Onboard clocks will, in general, be less accurate than clocks used in ground OD systems, and it will be necessary to account for timing errors. The errors are usually modeled as a bias and a drift, and the two associated parameters are included in the estimated parameter list.

A final point concerns the importance of the spacecraft attitude in the OD process. In many cases attitude determination and control are completely independent of OD. However, there may be requirements (e.g., to maintain a certain pointing accuracy continuously), that would couple attitude and orbit determination. Furthermore, if atmospheric drag contributes significantly to the spacecraft motion, then attitude and orbit determination are coupled, since the drag is a function of the spacecraft attitude (unless the spacecraft can be considered spherically symmetric).

REFERENCES FOR TOPIC 2.2.6

1. Tapley, B. D. and B. E. Schutz, "A Comparison of Orbit Determination Methods for Geodetic Satellites," in The Use of Artificial Satellites for Geodesy and Geodynamics, ed. G. Veis, Athens, 1973.
2. Kalman, R. E., "A New Approach to Linear Filtering and Prediction Problems," Journal of Basic Engineering, Vol. 82, 1960, pp. 35-45.
3. Battin, R. H., Astronautical Guidance, McGraw-Hill, New York, 1964, pp. 338-339.
4. Carlson, N. A., "Fast Triangular Factorization of the Square Root Filter," AIAA Journal, Vol. 11, 1973, pp. 1259-1265.
5. Bierman, G. J., Factorization Methods for Discrete Sequential Estimation, Academic Press, New York, 1976.
6. Thornton, C. L. and Bierman, G. J., "A Numerical Comparison of Discrete Kalman Filtering Algorithms: An Orbit Determination Case Study," NASA/JPL Technical Document 33-771, June 1976.

2.2.7* Propagate the Ephemeris

In an autonomous navigation subsystem the trajectory subsystem will have several functions: orbit prediction, ephemeris generation, event prediction, and maintenance of constants required for navigation algorithms.

2.2.7.1 Orbit Prediction. Orbit prediction consists of taking an initial spacecraft state vector at a specified time and propagating the trajectory to a second time to obtain a final spacecraft state vector. The trajectory subsystem is available to the remainder of the navigation subsystem for serving this function. The propagation algorithm may be analytic in form or involve the numerical integration of a set of equations modeling the force field. Orbit prediction falls into two major categories: special perturbations and general perturbations. Special perturbations refer to the prediction to an orbit by numerical integration in which the departure from a two-body solution is calculated. This is most useful for orbits of limited duration. Among the variety of methods available are Cowell's method (integration of total acceleration), Encke's method (integration of coordinates) and variation of parameters method. General perturbation methods comprise the analytical integration of series expansions of the perturbative accelerations. The criteria for selection must include state vector accuracy over a required time interval, the time available to perform the calculations, the range of possible mission types for which the system must be applicable, the word size of the computer being utilized, and the storage requirements. Simpler models may be possible if periodic updating of model parameters is allowed. Obviously the time period between model updates must comfortably exceed the duration requirements of autonomous operation.

2.2.7.2 Forces. A subset of the following forces acting on the spacecraft will need to be modeled: atmospheric drag, solar radiation pressure, non-spherical gravitational potential of central body, third body gravitational perturbations, gas leaks, thrusting maneuvers. The subset will be selected on the basis of size of orbit perturbation relative to orbit control parameters. For geosynchronous satellites with tight station control it may be possible to express the force model as a small perturbation on a perfectly geosynchronous satellite mission. This will simplify the model formulation and increase the accuracy of the calculations for the same computer word size.

In many implementations it is likely that the orbit determination subsystem will require the state transition matrix between any two selected epochs in addition to the state vector. Again this may be provided by analytic formulations but more probably it will require numerical integration of the variational equations. This section will comprise the bulk of the trajectory subsystem in terms of computing time and storage requirements.

2.2.7.3 Coordinate Systems. Central to any propagation scheme is the selection of the coordinate system in which the state vector and state transition matrices are expressed. Cartesian coordinates, relative to the

*By S. J. Kerridge

center of the primary body, allow relatively simple formulation of the accelerations and variational equations but have several disadvantages, including the wide range of values taken on by all coordinates, the relatively time consuming process of numerical integration and the relatively large algorithm implementations. A set of orbit elements, such as classical or equinoctial for instance, may be more appropriate for an onboard system that requires minimizing algorithm size. Approximation (such as small angle and first order perturbations only) should be made wherever possible to reduce the load on the autonomous system, provided, of course, that performance can be maintained. There is an obvious tradeoff implied here between the range of missions for which a particular autonomous system is appropriate and simplicity of a system applicable only to one specific mission or mission type. For example, a trajectory subsystem required to serve all Earth orbital missions will be more complex than one required to serve only geosynchronous satellite missions which, in turn, will be more complex than one only required to serve a geosynchronous satellite located at, say, 139°W latitude.

2.2.7.4 Celestial Body Ephemeris. Ephemeris generation involves the calculation of the coordinate of any appropriate solar system or celestial body at a specified epoch. In particular, for Earth orbital missions this will almost certainly include the Sun and the Moon which can, among other things, act as perturbing gravitational sources, as radiation sources and as possible objects to be observed by onboard sensors for attitude control and/or orbit determination.

As with orbit propagation, the complexity of the ephemeris algorithms is related to the accuracy and speed requirements placed upon this part of the trajectory subsystem. The algorithms will probably take the form of polynomials. The simpler ones could have time as the independent variable in a simple polynomial while more complex schemes may involve Chebyshev polynomials. The longer the time requirement over which the algorithm must perform adequately (in terms of accuracy) the more complex the formulation that will be required, and the more time-consuming will be the execution.

2.2.7.5 Event Prediction. Related to ephemeris generation and orbit propagation is the function of event prediction. Events such as solar occultation may have a major impact on the operation of a spacecraft as the need for spacecraft systems to be reconfigured when such an event occurs. The trajectory subsystem is responsible for predicting the occurrence of such events and making this data available to the appropriate spacecraft systems. The predictions are made by combining spacecraft position and celestial body ephemeris data. This will include quantities such as the angle at the Sun between the Earth and the spacecraft for predicting solar occultations.

2.2.7.6 Other Considerations. It is also assumed that the trajectory subsystem will be responsible for the maintenance of all parameter values and constants required for the autonomous navigation subsystem. This will avoid conflicts in responsibility. The parameters and constants appear in the algorithm formulation. A specific algorithm and associated parameters have a limited durability in terms of sufficiently accurate performance. The parameters will need to be updated at the conclusion of each of these time periods to preserve the accuracy for the succeeding time period. If this

updating is not done sufficiently often then performance will degrade. The degradation may be gradual or catastrophic depending on the formulation selected. In most cases the former would appear to be preferable.

An overriding concern of the trajectory subsystem, as of the navigation subsystem as a whole, is that it not get itself into a state from which it cannot recover. Algorithms should not be used which have questionable stability or restricted applications within the set of possible states which it will encounter. Reassurance in this regard is obviously tied to the adequate validation of the system. Prior to flight, validation will test the software to be used in the flight hardware with simulation of sensor data, maneuvers, etc. Validation will proceed at two levels for an operational system - the initial checkout phase and the routine operations phase. The latter will involve minimal ground monitoring and will not be essential to the adequate performance of the trajectory subsystem within the autonomy requirements, but will allow for possible augmentation of that performance if so desired.

2.2.8* Maneuver Planning

2.2.8.1 Functional Description. The Maneuver Planning function is required as a part of an Autonomous Navigation subsystem whenever the orbit of the spacecraft must be controlled. As indicated in Figure (page III-75) the function fits logically between the Orbit Determination segment where the current orbit state is estimated and the Maneuver Command segment where maneuvers are implemented.

The top level flow of activities within the Maneuver Planning segment is illustrated in Figure III-12. The process begins with an orbit estimate from the Orbit Determination segment. This estimate is then compared to the orbit established by the mission requirements. If orbital corrections are required then continued computations are needed. If not, the control is returned to the Executive program.

When it is determined that a maneuver is required then algorithms are executed which compute the changes necessary to accomplish the required orbital correction. The results of these computations are, typically, a maneuver time, magnitude of the velocity change, and the direction of the velocity change. Since the process does not, in general, account for all of the details of the implementation process, the output is described as the "desired" or "ideal" maneuver. Many algorithms will compute two or more desired maneuvers for a single orbital correction.

Not shown in Figure III-12 are the interfaces with the Trajectory segment. In order to reduce the computer requirements and maintain internal consistency, the Maneuver Planning segment should utilize the services of the Trajectory segment whenever Trajectory propagation is required.

The process of computing the desired maneuver depends upon the type of orbital correction required, the characteristics of the spacecraft and the orbit characteristics.

*By J. A. Kechichian

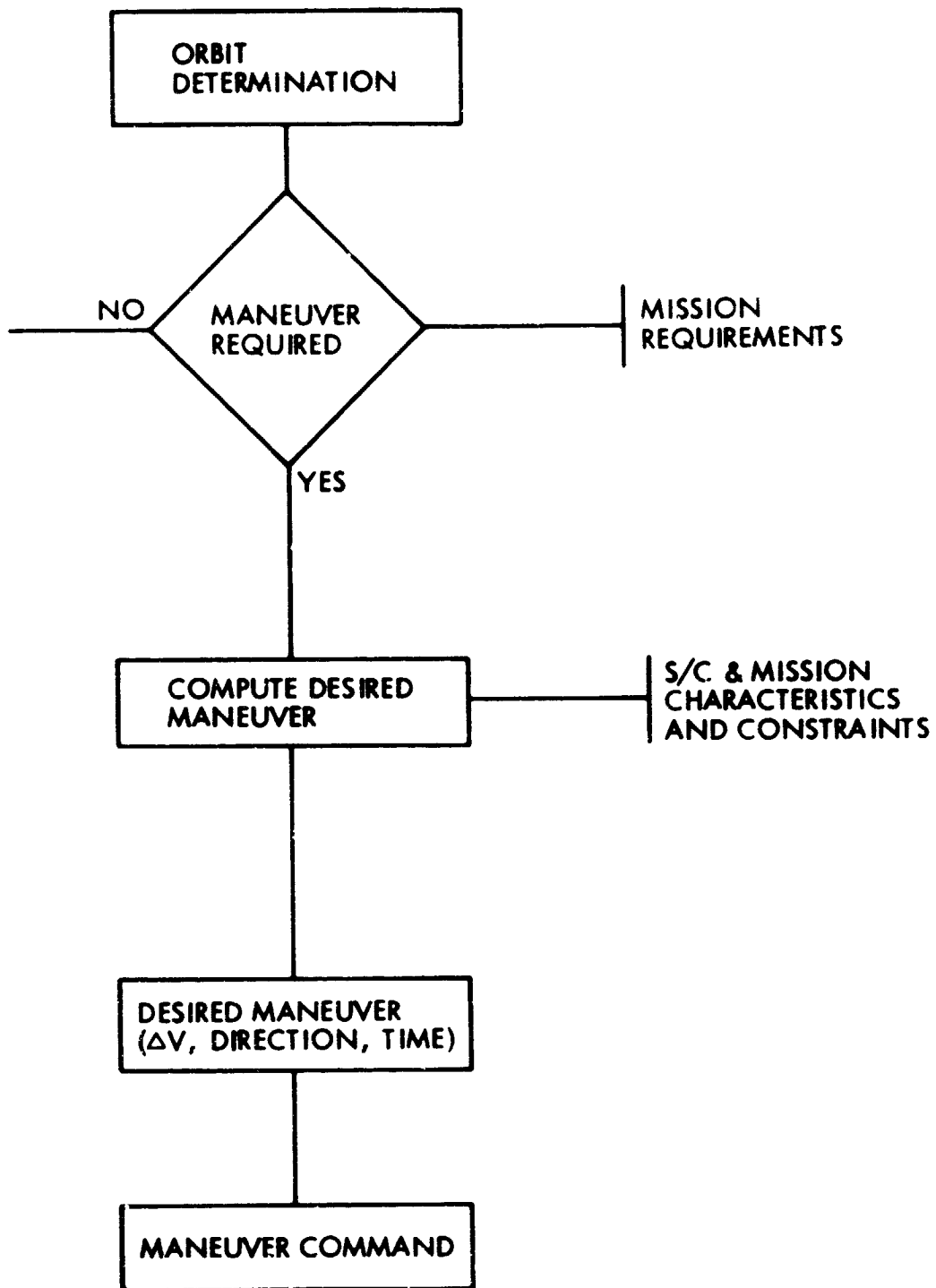


Figure III-12. Maneuver Planning Flow Diagram

The next section discusses the requirements on the Maneuver Planning function and the following section discusses the various techniques which might be employed. The most significant factor which should be clear from these discussions is the strong dependence of the Maneuver Planning function on both the mission and spacecraft characteristics. This implies that relatively little carry-over can be expected from one application to the next.

2.2.8.2 Requirements on Maneuver Function. The selection and implementation of autonomous maneuver algorithms for a particular application depends upon the requirements generated by:

- (1) The mission requirements
- (2) The operational characteristic
- (3) The architecture of the on-board autonomous system
- (4) The characteristics and operational procedures of the ground control system

The mission places a number of different types of requirements on the maneuver planning function. First and foremost, the mission requirements determine which elements of the spacecraft's orbit must be controlled and to what accuracy. For example, geosynchronous missions generally require that the spacecraft's position over a specified Earth-fixed longitude be controlled to within a specified tolerance in both latitude and longitude. A low Earth orbiter may require that the altitude be controlled within a specified deadband. The mission requirements may also place constraints on the maneuvers. The typical constraint involves the interaction of the maneuver with payload functions. Since on many spacecraft the execution of a maneuver impacts or requires temporary cessation of the payload function, there is a strong requirement to either minimize the number and/or magnitude of the maneuvers or restrict their locations.

The spacecraft operational characteristics and constraints also place significant requirements on the Maneuver Planning function. Since the Propulsion and Attitude Control subsystems are the principal subsystems involved with the execution of a maneuver, the characteristics of these two subsystems have the major impact. The Propulsion subsystem characteristics of primary importance are:

- (1) Propellant availability: If the propellant supply is very limited then the maneuver algorithms must utilize the propellant in the most efficient manner possible.
- (2) Thrust Characteristics: The time required to deliver a specified impulse, the characteristics of the magnitude control and the subsystem accuracy are all of importance. If the time required to deliver the required maneuver magnitudes is short relative to the period of the orbit, then certain

simplifications can be made in the maneuver algorithms. However, if the time is long, then more complex calculations are required. In this latter case, the characteristics of the acceleration magnitude may also influence the maneuver design. The maneuver design for long maneuvers using a "blow down" Propulsion Subsystem where the thrust magnitude decreases during the maneuver, is significantly different than the design for a high impulse system.

The characteristics of the thrust cut-off techniques may place constraints on the magnitudes of the maneuvers which can be achieved. Systems with accelerometer cut-off can typically deliver almost any maneuver magnitude, whereas systems with timer cut-off may only be able to deliver maneuvers in fairly large increments. Large quantization of maneuver magnitudes must be accounted for in the maneuver algorithms. Finally, systems with large error characteristics may require special techniques.

The characteristics of the Attitude Control subsystem also influence the maneuver design in a number of ways. Three-axis stabilized spacecraft and spinning spacecraft have very different maneuver characteristics. Three-axis stabilized spacecraft come in several flavors. Some are able to turn and point the acceleration vector in arbitrary directions whereas others maintain a fixed attitude and deliver the maneuver vector along specified directions (in components). There are also differences between spacecraft which are Earth-referenced and those which are inertial.

There are also some variations in spinning spacecraft. Some maintain a fixed pointing direction while other are able to reorient the spin vector to arbitrary directions.

The maneuver algorithms may be influenced by some or all of these spacecraft characteristics and constraints, and careful analysis is required before the design is finalized.

The third major driver on the Maneuver Planning functions is the architecture and requirements of the onboard autonomous system. Maneuvers must be planned in such a way as to fit within the overall autonomous control flow. Examples of requirements which might be generated in this case include interface protocol with other subsystems, execution time windows, and computation restrictions. One of the most significant requirements placed on the Maneuver Planning function within an autonomous navigation system is the requirement to produce "fail-safe" results. The algorithms must be designed to produce the correct results under all conditions. In particular the algorithms must be able to recognize erroneous results and take the appropriate action. Conditions which must be considered include bad input data, orbit conditions outside of the acceptable limits, and changes in the spacecraft configuration.

The final area placing requirements on the Maneuver Planning function is the ground control system. Since maneuvers are generally significant spacecraft and mission events, they must be planned in such a way as to allow complete ground system visibility and control. If at all possible the maneuvers should be planned such that ground control has the opportunity to review and possibly modify the plan before execution. It may also be

desirable to employ common techniques and algorithms in the spacecraft and on the ground.

In summary, all of the areas discussed above have the potential for placing requirements on the Maneuver Planning function. A careful analysis of requirements must be made before the design is finalized.

2.2.8.3 Maneuver Techniques. The techniques used to control the orbit of the spacecraft may be broken down into a number of general categories. They are:

1. Transfer
2. Rendezvous
3. Stationkeeping
4. Orbit Maintenance

The techniques used to generate maneuvers for each of these categories are also a function of the type of orbit (circular or elliptical), the requirements and constraints (minimum fuel, for example), the computation capability of the spacecraft, and the autonomous operations (required convergence). In the following paragraphs the techniques appropriate for each of the categories will be discussed and appropriate references given where possible. For current purposes only maneuver techniques applicable to "high" thrust spacecraft will be considered. There is another large body of techniques appropriate to spacecraft with "low" thrust spacecraft ion drive systems, for example).

2.2.8.3.1 Transfer. A transfer is defined as a general change of the position and velocity of the spacecraft meaning a general change from the initial kinematic conditions at time t_0 (r_0, V_0) to the final conditions at time t_f (r_f, V_f). A transfer can be totally specified (rendezvous) or partially undetermined (for example in the case of an interception, V_f is free). The location of the spacecraft in the final orbit is not normally controlled by the transfer solutions. In general a transfer is achieved in a certain optimum way, the most common criterion being the one that leads to the minimum fuel expenditure. Minimum time transfers, on the other hand, have obvious applications for rescue and interception missions. In either case it is necessary to choose the reference of firing locations along the orbit where the impulsive velocity changes have to be applied, with the appropriate thrust orientation and magnitude at each such location. It is assumed that the attitude control subsystem can freely orient the thrust vector in any desired direction in space. However, for spacecraft that are inertially fixed in space, this reorientation may not always be possible, in which case the transfer is of the constrained type.

The transfer solution is dependent on the type of propulsion subsystem used: the propulsion subsystems may be classified into two main categories: classical or chemical propulsion subsystems capable of relatively high thrust and imparted acceleration but relatively low specific impulse, and electric propulsion subsystems characterized by low thrust and high specific impulse. The first type leads to transfers that can be modeled by a sequence of finite impulsive velocity changes applied instantaneously, while the electric subsystems (ion drive) require continuous thrust programs to achieve the desired change of orbit. The impulsive transfer has received

a great deal of attention from several contributors who focused mainly on fuel minimizing solutions. Hohmann, Hoelker and Silber used the method of parametric optimization where the total number of impulses (in general 2 or 3) is fixed a priori, thereby reducing the investigation to the determination of the firing locations and thrust magnitude and orientation of each such location in order to minimize the arithmetic sum of the velocity changes resulting in a minimum fuel expenditure. This led to the well-known two-impulse Hohmann transfer and the three-impulse bi-elliptical or bi-parabolic transfer of Hoelker and Silber, between coplanar circular orbits. This method of parametric optimization cannot specify the optimal number of impulses for a given transfer and is inadequate for low thrust trajectory optimization. The application of the methods of functional optimization using Pontryagin's maximum principle has led to the successful solution of many types of transfer.

The transfer between neighboring near-circular orbits has received the most attention because of its many useful applications in Earth orbit. The solutions obtained by Contensou¹ (coplanar) and Marec² and Edelbaum³ (noncoplanar) are linearized optimal transfer solutions given in closed form and require at most two impulses to achieve the orbit change.

The general transfer between elliptic orbits is obtained through numerical iterations by Small⁴ and is rather difficult to implement onboard the Navigation system which must be fail safe, meaning that it must be capable of finding a solution for every possible situation. These iterative techniques usually require large computational capabilities, unlike the linearized closed form solutions which have the additional advantage of always providing a successful outcome.

It is sometimes important to consider simple non-optimal strategies which are much easier to implement, such as the Lambert type of transfers which consist of two impulses applied at the end points of the transfer orbit. The first impulse transfers the spacecraft on an orbit that intersects the final desired orbit, at which point a second impulse is applied to enter the final orbit. Finally, in the case of intersecting orbits, a single impulse applied at the point of intersection is sufficient to achieve the desired transfer.

2.2.8.3.2 Rendezvous. Rendezvous is an extension of the transfer problem where the location of the spacecraft in the final orbit is also controlled. The analysis of orbital rendezvous trajectories is in general more difficult than the transfer problem because of the excessive dimension of the state vector. A rendezvous problem begins with a vehicle performing a prescribed motion as the initial condition and ends with the vehicle performing a time-related prescribed motion as the final condition. There are mainly two types of rendezvous: the terminal phase rendezvous and the orbital rendezvous. Terminal phase rendezvous is specifically concerned with the relative motion between the passive target and the active rendezvous vehicle. The motion is restricted to small displacements with the equations linearized about the target body. Furthermore, a rendezvous may be achieved with a fictitious target, not necessarily a vehicle in orbit; an important example is the rendezvous in longitude for a geosynchronous spacecraft. Within orbit rendezvous, three important classes arise, namely the time-free, time-limited and time-fixed rendezvous.

The time-free rendezvous solutions degenerate into the corresponding transfer solutions with appropriate waiting periods between impulses, meaning the parking in intermediate ellipses obtained through the various impulse splitting techniques. Time-limited rendezvous usually involves a tradeoff between velocity change magnitude and a time, while time-fixed rendezvous determines the optimum trajectory between orbiting vehicles satisfying given initial and final times.

Unlike the transfer solutions, the minimum velocity change magnitude rendezvous solutions may require up to six impulses obtained through complex iterative routines. Such an example consists of the rendezvous between neighboring vehicles in noncoplanar near-circular orbits.⁵ The general problem of the rendezvous between general elliptic orbits is not yet solved or fully understood; nevertheless Lambert type solutions consisting of two impulses applied at the end points of the transfer orbit may be easily calculated. These solutions are, in general, nonoptimal. Considerable fuel may be saved by a two-parameter search on the end times without excessively taxing the autonomous onboard computational capabilities.

Finally, there is a class of special purpose rendezvous techniques. The primary example is the co-elliptic rendezvous technique developed and proven during the manned Gemini program and utilized during the Apollo and Skylab programs. These techniques involve preselected terminal approaches preceded by a sequence of maneuvers which yield the desired initial conditions. To date such techniques are limited to circular orbit operations.

2.2.8.3.3 Stationkeeping. Stationkeeping arises during missions involving satellite inspection and crew rescue on one hand, and during geosynchronous missions where the spacecraft is required to remain inside a narrow longitudinal deadband at the equator centered about a fixed ground station longitude. Stationkeeping therefore consists of keeping the spacecraft in the vicinity of a fictitious or real target once a rendezvous is achieved. For satellite inspection or prior to docking, a standoff position is often desirable. This is a location in the target vicinity from which operations required during the mission can be carried out. An ideal relative position with respect to fuel consumption is one which requires no active thrusting to maintain that position. This can effectively be the case if the spacecraft is lying on the orbit of the target vehicle with a small angular separation. A small radial impulse would force the spacecraft to oscillate about its neutral position allowing inplane inspection of the target in two dimensions. The East-West stationkeeping of geosynchronous spacecraft consists of an active control of the drift in longitude experienced by the vehicle under the influence of the Earth's tesseral harmonics, solar radiation pressure and luni-solar attraction. Several targeting strategies may be considered by the Navigation subsystem of the spacecraft. The drift cycle may be effectively repeated by controlling only the semi-major axis if the daily libration in longitude induced by the solar radiation pressure which affects eccentricity does not exceed the given tolerance. Otherwise both semi-major axis and eccentricity may be controlled by using the linearized transfer solutions for coplanar near-circular orbits mentioned in 2.2.8.3.1 above,

obtainable in closed form and easily implemented onboard the Autonomous Navigation subsystem.

2.2.8.3.4 Orbit Maintenance. Orbit maintenance consists of maintaining some or all of the orbit elements or even some parameters which are functions of these elements within specified tolerances, depending on the characteristics of the mission flown. This is achieved by maneuvering the spacecraft at regular intervals in order to correct for the effect of perturbations (geopotential, third body, solar radiation pressure, atmospheric drag) that affect the size, shape and orientation of the orbit and ultimately violate the specified deadband tolerance that results from the drift.

For example, it may be desired to control the period of a spacecraft in a 24-hour elliptic orbit by adjusting the semi-major axis by way of small impulse velocity changes applied at perigee.

Another example consists of maintaining a frozen orbit by keeping the argument of periapse close to 90° and effectively cancelling the effect of J_2 and J_3 zonals on eccentricity which then stay very close to zero.

Another important example consists of maintaining the nodal drift within a specified narrow deadband in order to obtain a ground track repeat at regular intervals (N day repeat orbit) for purposes of observation. The nodal drift is primarily the result of atmospheric drag for low-earth orbiters and can be effectively controlled by semi-major axis adjustment.

In many cases it may be necessary to design appropriate targeting strategies that allow the spacecraft to remain inside the tolerant deadband for the longest possible time, thereby minimizing maneuver frequency and fuel expenditure. It may therefore be necessary to consider the various targeting strategies in the control flow logic of the Navigation subsystem in order to carry out the orbit control more effectively.

REFERENCES FOR TOPIC 2.2.8

1. P. Contensou, Etude Theorique des Trajectoires Optimales dans un Champ de Gravitation: Application a un cas d'un Centre Unique. Acta Astronautica 8, pp. 134:150, 1962.
2. J. P. Marec. Transferts Impulsionnels Economiques Entre Orbites Quasi-circulaires Proches non Coplanaires, Acta Astronautica, Vol. 14, pp. 47-55, 1968.
3. T. N. Edelbaum, A General Solution for Minimum Impulse Transfers in the Near Vicinity of a Circular Orbit. Analytical Mechanics Associates Inc. Cambridge, Massachusetts.
4. H. W. Small, Minimum-fuel Time Free Transfer Between Elliptic Orbits. PhD Thesis, Stanford University 1972.
5. J. B. Jones, Optimal Rendezvous in the Neighborhood of a Circular Orbit: The Journal of the Astronautical Sciences, Vol. XXIV, No. 1, pp. 55-90, Jan-March 1976.

2.2.9* Maneuver Command Function

2.2.9.1 Functional Description. The implementation of maneuvers onboard a spacecraft in the final analysis involves sending a series of commands to various spacecraft subsystems. These subsystems almost always include Attitude Control and Propulsion and may also include Power, Telemetry and Communications. The function of the Maneuver Command segment of the Navigation subsystem is to transform the "ideal" maneuvers generated by the Maneuver Planning segment into the appropriate set of maneuver commands.

2.2.9.2 Interfaces. Before investigating the details of the Maneuver Command function, it is important to understand the interfaces. The Maneuver Command function is initiated after the Maneuver Planning function has decided that one or more maneuvers are required and has computed the time, magnitude, and direction of the maneuver.

This input is generally referred to as the ideal maneuver, since it is a result of numerical calculations and does not account for many of the realities of the spacecraft implementation. For example, the magnitude of the maneuver is defined only by the limits of the computer word length while the spacecraft may be only able to implement maneuvers with a coarser increment size. The same limitations may also be true of the maneuver time and direction. The Maneuver Planning segment may only check to see if the maneuver is feasible.

At the other end of the Maneuver Command function is the actual spacecraft. The commands must account for both the operational characteristics and configuration of the other spacecraft subsystems involved in the execution of a maneuver. For example, if the propulsion subsystem can only implement maneuvers in increments of a specified impulse size, then the magnitude of the ideal maneuver must be adjusted to fit. The Maneuver Command segment must also adapt to changing spacecraft configurations. For example, if attitude control is unable to implement the nominal control modes for a maneuver, then alternate appropriate modes must be selected. The output characteristic of the Maneuver Command subsystem are thus clearly dependent upon the spacecraft operational characteristics.

2.2.9.3 Detailed Description. It is for this latter reason that the Maneuver Command function must, ideally, execute the following actions. These actions are discussed in more detail in the following paragraphs.

1. Determine the appropriate control mode.
2. Transform the ideal maneuver parameters into commanded maneuver parameters.
3. Formulate the command sequence required to a) configure the spacecraft for the maneuver, b) execute the maneuver, and c) reconfigure post-maneuver.

4. Issue the commands at the appropriate time relative to the required maneuver time.
5. Place the necessary documentation parameters on the telemetry stream and audit trail.
6. Inform other segments of the Navigation subsystem of the commanded maneuver parameters.
7. Monitor the execution of the maneuver commands and issue abort commands if required.
8. Upon maneuver completion notify the remainder of the Navigation subsystem of any known deviations from the commanded parameters.

Determining the appropriate control mode is a function of the spacecraft design and current status. It is not unusual for a spacecraft to have two or more modes for executing a given maneuver. These modes must be prioritized according to some criterion (minimum fuel, simplicity, safety, etc.) and ground rules established for choosing a particular mode. In addition, the system must have knowledge of the hardware and/or spacecraft configuration required in order to utilize a particular mode. Using the ideal maneuver and knowledge of the spacecraft status, the Maneuver Command segment sorts through the potential modes in priority order until an acceptable mode is found. Of course, if an acceptable mode is not found, then the system aborts the maneuver and awaits ground instructions.

Given the control mode, the Maneuver Command subsystem can now adjust the maneuver parameters to match the capabilities of the system. As noted earlier, a typical example is adjusting the maneuver magnitude to an integral number of pulses. The adjustment procedure would follow a predetermined criterion. Typical examples which have been used for the latter are: minimum error, minimum change and next lower step. The results of the adjustment are used to determine the commanded maneuver parameters in action 6.

The third action is to formulate the command sequences. Since the autonomous features must operate in concert with the ground control function, the appropriate technique is to pre-construct tables which exactly parallel the ground command sequences. Given the control mode, the appropriate subset of sequences can be selected and the values determined in the previous action inserted into the command sequence. The command sequences must also contain auxiliary data which specifies the interrelationship between sequences, particularly relative to timing. It may also be useful to carry constraint data along with the command sequences. In order to minimize memory storage requirements a multi-level structure is probably appropriate.

Knowing the required command sequence and the maneuver time, the Navigation subsystem can now issue the commands at the appropriate clock times. The command sequence naturally separates into 3 blocks: configuration, execution, and reconfiguration. The configuration commands, which may begin an hour or more before the actual maneuver, bring the spacecraft to a readiness

state. For simplicity, the Navigation subsystem would issue high level commands with the directed subsystem issuing the detailed commands. Each step is monitored and the next step initiated only after verification of satisfactory execution of the previous step. The execution sequence both initiates and terminates the actual maneuver. Monitoring on several levels is appropriate. During the actual maneuver, abort commands may be issued by any of the subsystems involved with the maneuver (or, of course, the ground). After the maneuver the reconfiguration commands return the spacecraft to the desired "cruise" configuration. This may or may not be identical with the premaneuver configuration.

The Orbit Determination, Trajectory and possibly the Maneuver Planning segments of the Navigation subsystem will, in general, benefit from having knowledge of the commanded maneuver parameters. The Maneuver Command segment is the central distribution point. Relative to action 8, if the commanded maneuver is modified and/or aborted, then the Maneuver Command segment must pass this information along. The appropriate technique is probably a table of actual parameters which is updated by only this segment.

The overall development of this segment must carefully consider the requirements for fault tolerance. Special checks must be developed to insure the accuracy of the commands issued. Issuing command in a reverse sequence, for example, could create a spacecraft failure. Therefore, it will probably be appropriate to develop interlock sequences with the central executive controller to doubly protect against erroneous commands.

2.2.10* Verify Navigation Performance

A Navigation subsystem onboard an autonomous spacecraft must have the same level of failure detection, isolation, and correction capability as the other spacecraft subsystems. In a Navigation subsystem this implies the monitoring of the sensors, computer status, and algorithms. Since the output of the Navigation subsystem generally takes the form of data or commands operated on by other subsystems, care must be taken not to provide erroneous outputs.

The status of the sensors must be monitored continuously in order to insure that "bad" data is not introduced into the orbit estimation process. When errors are encountered, steps must be taken to either recalibrate or switch to alternate data sources. In some cases, sensors may be shared with the Attitude Control subsystem, in which case the proper protocol must be firmly established. Detecting sensor failures is improved by the availability of either redundant data or redundant data sources.

The techniques used for monitoring and correction of computer hardware problems will be a strong function of the overall architecture. For example, if the navigation function shares the computer facility with other functions, then the computer failure protection may be handled by an executive program and the navigation subsystem monitors only those portions of the computer under its exclusive control. In this case monitoring for bit errors may be sufficient.

*By J. B. Jones

If, however, the navigation function operates in a dedicated computer facility, then additional hardware failure protection is required. The external monitoring by the use of commanded test from a separate fault tolerant computer appears to be an attractive option.

The monitoring of the navigation algorithms may be both the most critical and most difficult of the failure detection functions. This function is currently executed by skilled analysts in the ground navigation centers. While a large body of knowledge exists relative to the problems which occur, little if any work has been accomplished toward translating this knowledge into automatically operating procedures.

2.2.11* Historical Summary of Autonomous Navigation Studies

The earliest paper which considered autonomous navigation appeared in the literature in 1963¹ and initiated a series of papers which considered the use of horizon scanners coupled with an inertial reference to autonomously determine the orbit of the spacecraft.^{2,3,4,5,6} Following this lead a number of papers began proposing the use of unknown landmarks for navigation purposes.^{7,8,9,10,11,12} Finally in the late 60's and early 70's three general studies were conducted by the Aerospace Corporation.^{6,13,14} These three studies considered a wide range of possible navigation measurements and provided a basis for judging their relative potential over a range of orbits.

All of the early studies were either covariance or Monte Carlo analysis and provided data on expected orbit determination accuracy as a function of the system configuration and assumed error levels. The general conclusion that could be drawn from this work was that autonomous navigation was a viable concept, but that the existing navigation sensors did not yield sufficient accuracy.

Beginning in the early 70's a broad range of activities was undertaken by the Air Force. One of the earliest and possibly the brightest spot in the entire history of autonomous navigation was the design, implementation and flight test of an autonomous stationkeeping system on the LES-8/9 Spacecraft.^{15,16,17,18} These spacecraft were in a 24-hour ecliptic orbit and the system was designed to automatically acquire and maintain the mean longitude within $\pm 0.15^\circ$ of the specified value. The system included all of the components required to determine the actual mean longitude and then to compute and execute the necessary orbit corrections. The LES-8 spacecraft was placed under active autonomous stationkeeping control during the period from 7 July to 4 October 1976. The flight results indicate that the spacecraft acquired and maintained a station at 109.7° W longitude with an absolute accuracy of 0.06° .¹⁸

While the LES-8/9 system was highly simplified and operated under rather special conditions, it did indeed demonstrate that autonomous navigation was a viable concept.

*By J. B. Jones

In parallel with the LES-8/9 flight experiment the AF initiated four sensor development activities. IBM developed a system which used the signals from known active radar stations as the basic measurement.^{19,20,21} This program proceeded through the critical component development phase and predicted accuracy levels ranging from 68 meters (1 sigma) for the Molniya 12 hr orbit to 900 meters for a synchronous orbit.

Honeywell and TRW concentrated on the development of special sensors in order to enable unknown landmark navigation. TRW developed an engineering model of a gimbaled tracker which provided tracking of stars and unknown landmarks.²² The system was carried through the lab test phase. The Honeywell system utilized two body fixed silicon matrix photo detectors to measure the landmark motion.^{23,24} This system carried the sensor assembly through the critical component development phase. Performance on the order of 1 km for low altitude orbits was expected.

Martin Marietta is developing the Space Sextant.^{21,25,26,27} This instrument, which employs two gimbaled trackers, is designed to measure angles between stars and either the lunar limb or Earth landmarks. Performance is expected to be in the 250 meter range. This development has proceeded further than the other three and is currently scheduled for a shuttle flight test.

More recently the AF has initiated development by TRW of a charge coupled device (CCD) Star tracker (the MADAN program).²⁸ This versatile accurate instrument is intended to provide the inertial reference required by an autonomous navigation system.

With the development of the AF Global Positioning System (GPS) both Aerospace Corp. and NASA have considered using GPS data to support an autonomous navigation system.^{29,30,31} NASA in combination with the Navy has developed a complete onboard package which has been tested and is currently being integrated into the Landsat-D Spacecraft. The system will be treated as an experiment during the mission and is expected to produce accuracy levels on the order of 10 - 20 meters. The estimated orbit will be used to annotate data from other onboard instruments prior to transmission to the ground.

In addition to these instrument development activities, a number of system studies have also been recently conducted. The British Aerospace Corp. conducted a study of a completely autonomous stationkeeping system for a geostationary satellite.³² The proposed system utilized Earth, Sun, and Star sensors to achieve complete control over the satellite station. In 1981 two papers reported on the results of a similar German study.^{33,34}

During this same time period the AF initiated the Autonomous Spacecraft Project (ASP) which included as one component an autonomous navigation system, with the added requirement of onboard fault detection and correction. This effort is proceeding towards a preliminary design by the end of FY82.

Attached is a reasonably complete Reference list of autonomous navigation studies.

REFERENCES FOR TOPIC 2.2.11.

1. Gunckel II, T. L., "Orbit Determination Using Kalman's Method", Journal of Institute of Navigation, 10, (3), Autumn 1963.
2. Meditch, J. S., Le Mary, J. L., Janus, J. P., "Analysis of an Horizon Scanner Autonomous Orbital Navigation System", Aerospace, TUK-46, (554U)-10)-6 (SSD-TR-65-102), June 1965.
3. Knoll, A. L. and Edelstein, M. M., "Estimation of Local Vertical and Orbital Parameters for an Earth Satellite Using Horizon Sensor Measurements", AIAA Journal, 3, (2), 338-45, February 1965.
4. Fitzgeralds, R. G., "Studies of a Horizon Scanner Autonomous Orbital Navigation System", Aerospace, TUK-0059(6311)-11, August 1970.
5. Hendrickson, H. T., "A Study of a Horizon Scanner Autonomous Orbital Navigation System", Aerospace, TOR-0059(6311)-11, August 1970.
6. Brogan, W. L. and LeMay, J. L., "Autonomous Orbit Determination at Synchronous Altitude," AAS 68-129, September 1968.
7. Levine, G. M., "A Method of Orbital Navigation Using Uperical Sightings to Unknown Landmarks", AIAA/IUN Guidance and Control Conference, August 1965.
8. Bellantoni, J. J., "Unidentified Landmark Navigation", AIAA Journal 5 (8), 1478, 1967.
9. Fischer, J. J. and Kochi, K. C., "Onboard Orbital Navigation Using Unknown Landmarks", Report No. TR-1855/311, Autonetics Division, North American Rockwell, Downey, California.
10. Hendrickson, H. T., "A Study of Unknown Landmark Tracking Navigation System, First Set of Results", Report No. TOR-0059(6311)-18, The Aerospace Corporation, El Segundo, California, January, 1971.
11. Hendrickson, H. T., "A Study of an Unknown Tracking Navigation System - Second Set of Results", Report No. TUK-0059(6311)-20, The Aerospace Corporation, El Segundo, California, January, 1961.
12. Hendrickson, H. T., "A Study of an Autonomous Unknown Landmark Tracking Navigation System", Report No. TR-0059(6311)-4, The Aerospace Corporation, El Segundo, California, February 1971; also presented at Institute of Navigation Meeting, Huntsville, Alabama, February 23-25, 1971.
13. Gura, I. A., Abbot, A. S. and Hendrickson, H. T., "Configuration Studies for Autonomous Satellite Navigation", Aerospace, SAMSU TR 71-166, May 1971.

14. Le May, J. L., et al, "High Altitude Navigation Study (HANS)", Aerospace, SAMSU TR-74-10, June 1973.
15. Srivastava, S. et al, "Design of an Autonomous Stationkeeping Controller for LES-8/9", MIT, ESU-TR-74-39, February 1974.
16. Srivastava, S., "Navigation Schemes for Les-8/9 Autonomous Stationkeeping System", MIT, AD-759-806, April 1973.
17. Srivastava, S., "Design of a Solar Ephemeris Synthesizer for LES-8/9", MIT, AD-752 250, October 1972.
18. Srivastava, S., "Flight Results of LES-8 Autonomous Stationkeeping System", MIT, April 1978.
19. Fritz, R. et al, "Autonomous Navigation Technology (ANT) System", IBM, SAMSU TR No. 74201, September 1974.
20. Aldrich, D. H., Simons, J. W., & Toda, N. F. "Self Contained High Altitude Navigation Systems Study: PRAIS Navigation System", IBM, SAMSU TR No. 77-57, January 1977.
21. Greenleaf, G. L., Mikelson, D., Aldrich, D., "Onboard Spacecraft Navigation Techniques", VII IFAC Symposium on Automatic Control in Space, 1976.
22. Williams, I. J., "Autonomous Navigation Technology - Phase I", TRW, SAMSU TR No. 74-244, August 1974.
23. Paulson, D. C., "Autonomous Satellite Navigation from Strapdown Landmark Measurements", Proceedings of the Third Symposium on Nonlinear Estimation Theory and Its Applications, San Diego, California, September 1972.
24. Paulson, D. C., "Autonomous Navigation Technology; Phase 1A Study", Honeywell, SAMSU FR-74-221, March 1975.
25. Serold, A. C., Bundy, L. N. and Schlemmer, J. W., "An Autonomous Satellite Navigation and Attitude Reference System", Proceedings of the ION National Aerospace Meeting, Martin Marietta, April 1977.
26. Booker, R. A., "Space Sextant Autonomous Navigation and Altitude Reference System: Flight Hardware Development and Accuracy Demonstration", Martin Marietta, Rocky Mountain Guidance and Control Conference, March 1978, AAS 78-124.
27. Mikeson, A. D., et al, "Space Sextant High-Altitude Navigation System (SS-HANS) Study", Phase U Final Report, SAMSU-TR-75-73, Martin Marietta, March 1975.

28. Lavery, M. P., et al, "Multi-Mission Attitude Determination and Autonomous Navigation (MADAN)," TRW, Annual Rocky Mountain Guidance and Control Conference, AAS paper No. 80-029, February 1980.
29. Farr, J. E., "Space Navigation Using the Navstar Global Positioning System (GPS)", Paper Number AAS-79-001, Annual Rocky Mountain Guidance and Control Conference, Keystone, Colorado, February 1979.
30. Fuchs, A. J., Wooden, W. H. and Long, A. C., "Autonomous Satellite Navigation with the Global Positioning System", AAS paper, September 1977.
31. Wooden, W. H. and Dunham, J. B., "Simulation of Autonomous Satellite Navigation with the Global Positioning System", AIAA Paper 78-1429, August 1978.
32. Vendy, B. and Plummer, C., "The Autonomous Stationkeeping System (ASKS)", British Aerospace, July 1978.
33. Eckstein, M. C. & Hechler, F., "Optimal Autonomous Stationkeeping of Geostationary Satellites", AAS paper No. 81-204, August 1981.
34. Leibold, A. and Eckstein, M., "Result of a Study of On-Board Autonomous Stationkeeping of Geostationary Satellites and its Impact to Ground Systems", AIAA/NASA Symposium on Space Tracking and Data Systems, June 1981.
35. Braga-Illa, A. A., "Automatic Satellite Stationkeeping", J of S/C & Rockets, Vol 6, April 1969.
36. Braga-Illa, A. A., "The Future of Self-Contained Control of Synchronous Orbits", MIT.
37. Brogan, W. L. & Le May, J. L., "Autonomous Satellite Navigation: An Historical Summary and Current Status", Paper 21-1, no date.
38. Brogan, W. L., "A Study of Autonomous Orbit Navigation Utilizing Known Landmark Tracking", Aerospace, TOR-1001 (2555)-3, December 1966.
39. Dunham, J. B., et al., "Algorithms for Onboard Orbit Estimation with Tracking and Data Relay Satellite System Data", AAS paper No. 81-204, August 1981.
40. Farrar, R. J., "Advanced Missions Operations Concepts, USAF/NASA Task 4, Vol. IV: Special Study, Survey of Autonomous Navigation Methods for NASA STS Advanced Mission", Report No. TOR-0059(6759-05)-1 Vol. IV, The Aerospace Corporation, El Segundo, California, July 1970.

41. Hand, J. A., "STAR/HORIZON Measurements for Onboard Spacecraft Navigation", MIT, April 1969.
42. Joseph, K., "Space Shuttle Horizon Sensor Navigation Analysis", Report No. 14492-6001-R0-00, TRW Systems Group, Redondo Beach, California, October 1970.
43. Markley, F. L., "Autonomous Satellite Navigation Using Landmarks", NRL, AAS Paper 81-205, August 1981.
44. Muller, E. S., and Kachmar, P. M., "A New Approach to On-Board Orbit Navigation", presented at Institute of Navigation Meeting, Huntsville, Alabama, February 23-25, 1971.
45. Shell, A. M. and Shenitz, C. M., "A Feasibility Study for a General Microprocessor-Based Orbit Determination System", Computer Sciences Corp., CSC/TM-79/6272, October 1979.
46. Toda, N. F., and Schlee, F. H., "Autonomous Orbital Navigation by Optical Tracking of Unknown Landmarks", AIAA/JACC Guidance and Control Conference, August 1966.
47. Toda, N. F., "An Autonomous Navigation Technology System", Proceedings of the Third Symposium on Nonlinear Estimation Theory and Its Applications, San Diego, California, September 1972.
48. Welch, J. D., Silvertson, W. E. and Wilson, R. G., "Landmark Tracking Technology", GE, AAS Paper, August 1975.
49. White, R. S., Adams, M. B., Geisler, E. G., Grant, F. D., "Attitude and Orbit Estimation Using Stars and Landmarks", IEEE Transactions on Aerospace and Electronic Systems VOL. AES-11, No. 1, March 1975.

2.3* DESIGN REQUIREMENTS FOR SUPPORT OF VALIDATION AND TESTING

The fundamental need to test and validate the proper operation of an autonomous spacecraft at the system and subsystem levels imposes requirements on its design. These requirements have been derived from experience with validation and flight operation of autonomous features and are grouped together without detailed discussion of the rationale or experience that has generated them. Provision of a record of autonomous control operation and associated status data, termed an "Audit Trail", is a design feature which has not yet been implemented. The complexity of autonomy for the overall spacecraft system suggests that provision of an audit trail will be a design requirement if the autonomous spacecraft is to be tested or operated with any reasonable degree of effort.

2.3.1 Validation Requirements on System Design

The following requirements may be satisfied by hardware design features, software implementation, or a combination. Recognition of these requirements early in the design process is important to allow maximum flexibility in implementation before it is limited by hardware constraints.

- (1) Spacecraft design shall provide for external initiation of autonomous control routines in a manner consistent with expected in-flight conditions.
- (2) Real time telemetry or stored audit trail data should provide information relative to cause of entry into a routine, action taken, and the status of the routine/and or resources under control of the routine.
- (3) Self-test capability should be provided to periodically exercise functions and ensure that recent operational status information is available. Such self-test operations should not interfere with normal mission operations.
- (4) Automatic 'safe-hold' modes shall be entered only when autonomous fault recovery criteria indicate that catastrophic conditions exist which prohibit reconfiguration and resumption of normal operations.
- (5) Fault recovery criteria shall be designed such that non-catastrophic events permit a fixed number of cycles through available redundant equipment, in an increasing time frame, before entering the 'safe-hold' mode if the repair action is unsuccessful.
- (6) Fault recovery criteria shall be established to limit toggling among failed elements.

*By H. R. Malm, J. Morecroft & R. Turner

- (7) Fault management processing shall be accomplished by a single software module within each subsystem to simplify the location of fault management logic. Error indications shall unambiguously identify the source of the error to the level of replaceable spares.
- (8) Flight system uplink and downlink data shall include a time tag indicating the command execution or data sampling time. The time tag shall be capable of being easily and unambiguously converted to Universal Time Constant (UTC). Time tag resolution should be sufficient to allow time localization of data as required by the spacecraft and the ground. Experience indicates that uplink data time tags should have a minimum resolution of 0.1 seconds and downlink data tags should have a minimum resolution of 0.001 seconds.
- (9) It shall be possible for ground control to inhibit autonomous functions and the execution of autonomous function output commands. The output command inhibit shall be automatically reset.
- (10) A self-test or checksum validation shall be performed for all command functions prior to event execution.
- (11) All commands that change the state of hardware or software operation shall unambiguously command a transition from one defined state to another. "Toggle" commands which alternately switch an operating state with repetitive executions of the same command word shall not be used.
- (12) Critical mission states and commands shall be periodically checkpointed such that fault recovery can roll back to a known event and re-establish the spacecraft state.
- (13) The flight system design shall provide data for alarm conditions, alert messages, errors detected, and nominal events.
- (14) Fixed periodic status messages shall be provided for all autonomous elements with command generation interfaces.
- (15) An unambiguous alert indication is required upon initiation of any autonomous function.
- (16) The mission ground systems shall provide an uplink command system design using techniques similar to the NASA Command Standard such as 'store and forward' and 'blind commanding' concepts so that ground controlled validation sequences can be uplinked rapidly from a primary command station and then enabled for execution from any available fixed or mobile command site.

- (17) Interface, cabling, and grounding design are subjected to close scrutiny to minimize system integration and test problems. Particular emphasis is placed upon separation of quiet and noisy circuits, implementation of the single point grounding tree philosophy, and detailed interface documentation. An Interface Control Document (ICD) is required for all element interfaces, in addition to any other documents bearing upon those interfaces.
- (18) The test hardware interface between ground support equipment and flight equipment must meet all design, environmental, and quality assurance requirements established for equivalent flight hardware with which the test hardware interfaces.
- (19) Where support equipment supplies power to flight systems and/or interfaces, the power sources must be regulated and isolated, with voltage and current limiting to prevent catastrophic overloads during the test program.
- (20) Power and signal grounds are to be isolated in all cases, with a common connection only at a single point in the ground tree configuration. Power connections to the spacecraft must be isolated so that a single fault of any power distribution line to the structure will not result in catastrophic power conditions.
- (21) All software modules or routines executing on the same processor or utilizing a common memory shall be designed with a special set of rules or protocols for buffer initialization and allocation to minimize data overwrite and transfer problems.

2.3.2 Audit Trail Requirements

2.3.2.1 Information Recording and Storage. Test, validation, and flight operations of autonomous spacecraft require the recording and storage of status and command information for later analysis. The audit trail must provide information relative to an event that allows unambiguous determination of the precursor condition(s) and the function operations, deviations from nominal performance, or periodic system/subsystem status snapshots. Types of information for inclusion in the audit trail are:

- (1) Time - An unambiguous time must be incorporated in all audit trail records to show the time of the first measurement of the first bit in the record. Time information must be easily converted to UTC and have a resolution of 0.001 seconds.
- (2) Latest Available Data (LAD) - The LAD from all sensors and critical derived measurements must be tabulated in a format which shows the last data value, the current data value, data suppression values, and performance criteria values for alarm, alert, event, and error conditions.

The LAD measurements would be included in the audit trail on initialization and when data suppression or performance criteria values are exceeded.

- (3) Fixed Periodic Status Data - Nominal operation within the limits of the LAD criteria is to produce an audit trail record at a fixed period interval to indicate system and subsystem status. The record is to summarize system elements as ready or not ready. Elements identified as not ready are to include a count of alarms, alerts, events, and errors detected since the last fixed periodic status interval.
- (4) Alarm Data - The violation of an absolute design limit is to cause immediate generation of a LAD record followed by an alarm record indicating the last value, the current value, and the alarm limit value which has been violated. The alarm record is to be repeated at each measurement sample period until the violation is corrected.
- (5) Alert Data - The violation of an operational limit is to cause immediate generation of a LAD record followed by an alert record indicating the last value, the current value and the alert limit value which has been exceeded. The alert record is to be repeated at each measurement's sample period in which a value change is detected. Should the measurement sample produce a value which is twice the critical alert value, the alert record is to be repeated until the violation is corrected.
- (6) Event Data - An event record is to be generated for a collection of N events or once every N minutes. This record is to indicate the last N events, the current event, and the alarm and alert status sampled at the current event time.
- (7) Error Data - The error record is to be a summary of syntax and processing errors which were detected and automatically corrected by error correction algorithms. This error record is to be generated for every M errors and is to show an unambiguous indication of the error source, the last error count, the current error count, and the error count criteria value.

2.3.2.2 Implementation Logic Requirements. Detailed analysis and planning of the audit trail content will be necessary to size the storage device, manage the available storage resource, and keep playback time to a reasonable level. The following requirements may be imposed on implementation logic:

- (1) An unambiguous audit trail shall be provided for all autonomous functions for the maximum period of unattended operations as defined in the mission plan.

- (2) Data compression and record frequency allocation techniques will be utilized to prevent excessive use of audit trail storage resources in normal operation.
- (3) Storage logic will allow for automatic deferral of non-critical events data if the storage resource is close to being filled in autonomous operation.
- (4) Techniques to suppress redundant data will be applied to all audit trail records.
- (5) Content and frequency of audit trail data shall be programmable after launch, and classes of events will be defined so as to be suppressible from inclusion in the audit trail by ground command or autonomous control action.

SECTION 3

SUBSYSTEM LEVEL AUTONOMOUS DESIGN CHARACTERISTICS

This section details the autonomous design characteristics of generic spacecraft subsystems. The information in this section is necessary to support architecture and control authority allocation trade-offs between spacecraft system level and subsystem level resources. The subsystem characteristics presented here are not exhaustive but represent the major subsystem functions that could require autonomous control to satisfy mission requirements. The material discussed in this section is generic in nature, and there is not sufficient design detail available to produce an actual implementation algorithm for the functions described. The intent is to identify potential autonomous functions and supply enough information to guide the spacecraft system and subsystem designers in selection of functions and provide a potential approach for implementation. The designers must consider their own constraints in the design of an individual implementation of any specific function. The specific control algorithm examples of Appendices B and C are to be used together with the generic design topics of Section 2 of Part III to aid the designer in this process.

Each subsystem characterization has three parts. The first discusses the nature of the subsystem functions and the functional elements that result in hardware implementation. The emphasis is to identify these items and their control characteristics that may be affected by autonomy. The second part is titled "Autonomous Maintenance Functions". Potential welfare maintenance functions are identified and characteristics are described in tabular form. Scope of the function's impact on the system/subsystem performance, subassemblies involved, and typical execution frequencies are listed. A candidate set of input, processing, and output requirements for a generic approach to implementation is presented. The third part performs a similar role for autonomous fault management functions. The particular characteristics listed for identified functions are the functional fault type, symptoms, impact and criticality, and a prospective example of the approach to detecting, isolating, and correcting the fault. Any flight project experience with protection algorithms for the fault is noted. Solutions for some faults are represented by example algorithms in Appendices B or C of this Handbook, and any that are applicable are referenced.

3.1* TRACKING, TELEMETRY, AND COMMAND (TT&C) SUBSYSTEM

3.1.1 Functional Description and Elements

The TT&C provides telecommunication functions for the spacecraft (S/C). The TT&C functional block diagram is shown in Figure III-13. This block diagram shows the three major functions; the Uplink, Tracking, and Downlink. The Antennas, Antenna Control, Antenna Select, Microwave Components and the Control and Monitor are functional elements of the TT&C used to accomplish the three key functions. These functions are described below in terms of functions and the elements which make up the function. The functional elements are depicted in the block diagram of the figures of this section. These elements are labeled "A....N" which denote a number of block or functionally redundant elements.

*By S. O. Burks

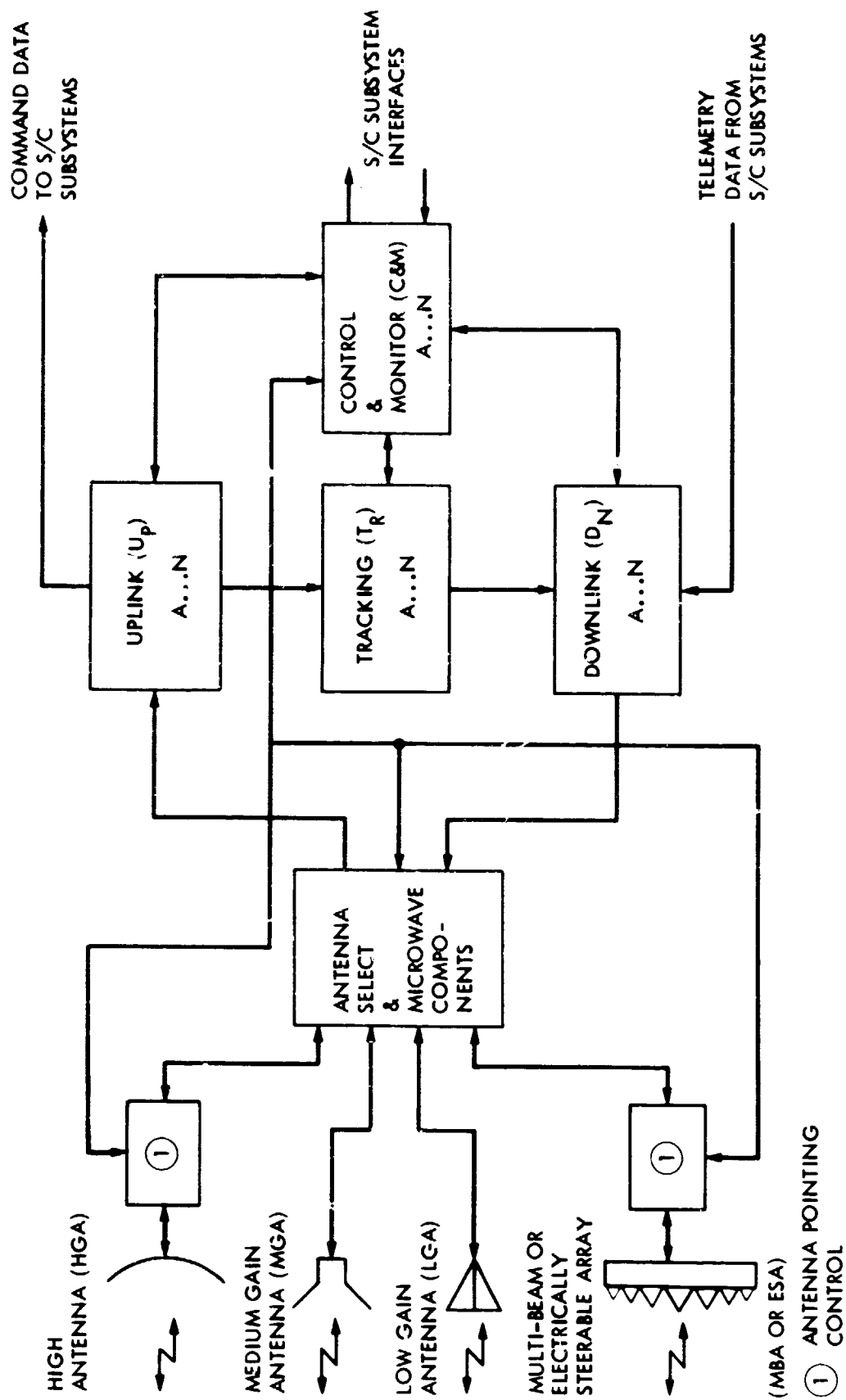


Figure III-13. Generalized TT&C Functional Block Diagram

3.1.1.1 Uplink Functional Elements. The uplink functional block diagram is in Figure III-14. An RF carrier from a ground, airborne or spaceborne source is transmitted to the satellite. The uplink function typically tracks the carrier and recovers the information modulated on the carrier. This information is in the form of commands or tracking data. The uplink function then stores and/or issues the commands and conditions the two-way data for transmission on the downlink function. The elements performing this function are as follows:

- (1) The antenna elements receive the RF signal. These antennas can be of several types from very broad "omni" antennas to narrow beam parabolic reflectors. As indicated in the Figure there can be one or several antennas and antenna types used to receive the RF signal. (Optical communications will not be assumed here although techniques similar to RF are used.)
- (2) Associated with the steerable antennas could be an electrical or electro-mechanical pointing system. These steerable antennas could be directed by automatically tracking the uplink signal (e.g., mono pulse or conscan), by preprogrammed pointing, or by using the satellite attitude for pointing.
- (3) The antenna select and microwave components provide for splitting, steering and filtering of the RF signal to the receiver.
- (4) The RF head and down converter filters the incoming RF signal and converts it to a lower frequency.
- (5) The IF, tracking and data demodulation element provides for signal amplification and filtering, tracking of the RF signals, and demodulation of data from the RF carrier.
- (6) The data detection element receives the demodulated signal and strips out the data (e.g., removes the binary coded command data from the command subcarrier).
- (7) The data handling element decodes the messages in the data (including decryption); stores some of the information for later use; issues preprogrammed stored commands; issues commands to the subsystems; and directs or takes direction for command action relative to autonomous functions. Some of the detected data (e.g., carrier tracking and ranging) is conditioned for downlink transmission as tracking data.
- (8) The data output selector is a buffer system for getting commands and tracking data from the data handling element to the satellite subsystems.

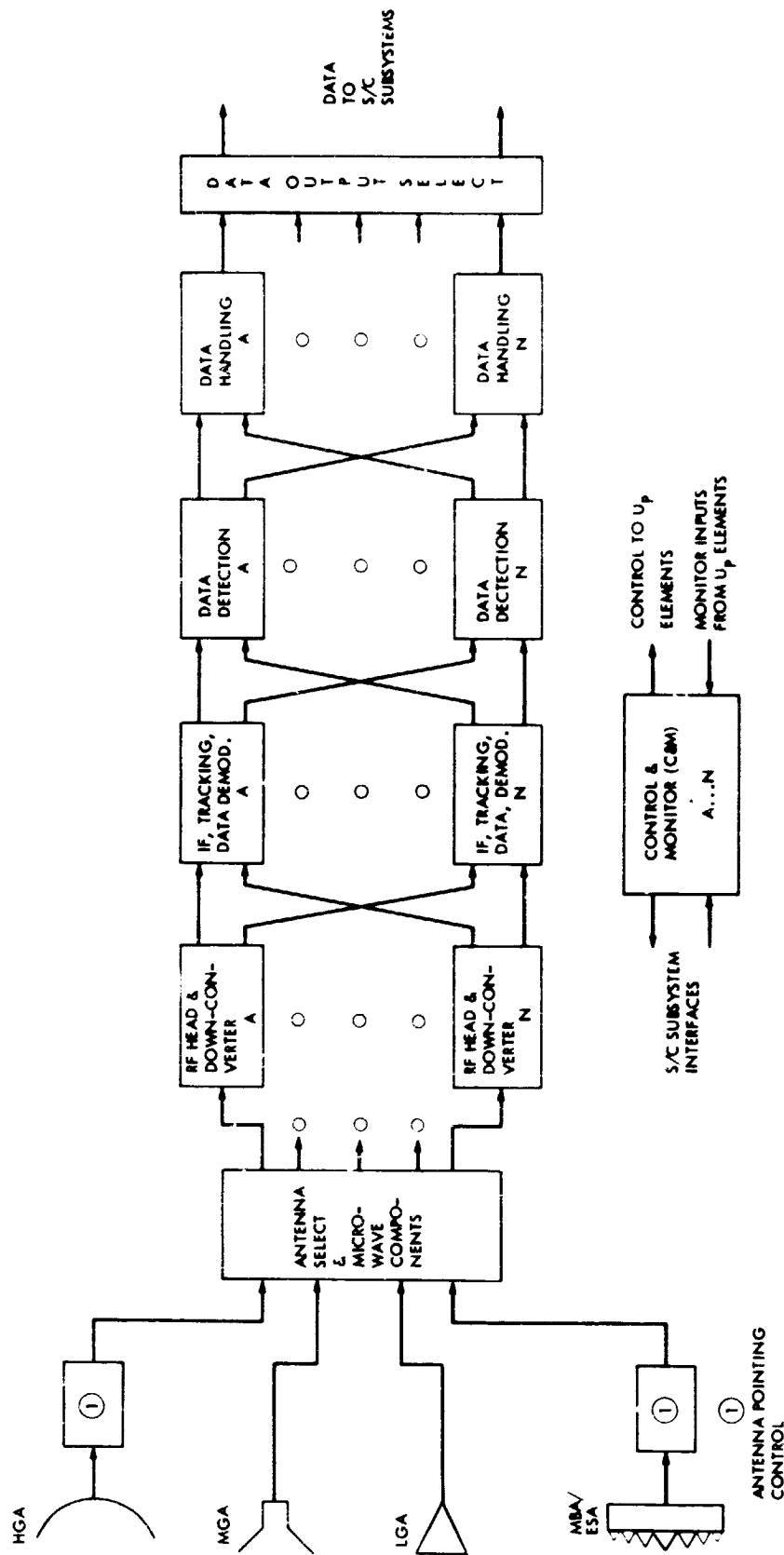


Figure III-14. Generalized TT&C Uplink (U_p) Functional Block Diagram

- (9) The control and monitor element provides for internal TT&C state changes and data conditioning. This element would be a key interface with other satellite subsystems, including autonomous interface features. This element would contain the distributed processing unit for the TT&C in a decentralized approach. The control feature of this element receives command inputs from the data handling system (initiated from the ground, by on-board storage or by the autonomy function). The control basically controls the mode and state of the various uplink functional elements. It also serves to transfer TT&C data to other subsystems, including data needed for autonomy; provides the interface for sensor data used for telemetry and fault detections; and provides and receives data to and from other subsystems.

3.1.1.2 Downlink Functional Elements

The downlink functional block diagram is in Figure III-15. The functional elements shown in the Figure provide for the transmission of data to the ground or to other air or space systems. The data, including autonomy-related data, is received from the satellite subsystems, and is then stored, conditioned, formatted and transmitted over an RF link. The elements performing this function are as follows:

- (1) The Input Selector is basically a data buffer between the data "sender" and the downlink function. It selects the data source(s) for use in the data handling system.
- (2) The Data Handling element receives the various forms of analog and digital data. The data is conditioned, formatted and possibly stored for transmission (e.g., Analog to Digital (A to D) conversion, coding, time division multiplexing, and storing for later transmission).
- (3) The Signal Conditioning element receives the data stream (typically binary coded) and conditions it for modulation (e.g., modulation on a subcarrier and/or adding to a pseudo random noise code). The signal conditioner may also accept the translated uplink signals for two-way tracking.
- (4) The Modulator and RF Driver receive the "conditioned" data and modulate it onto the RF carrier. The modulation can take many forms, e.g., amplitude modulation (AM), phase modulation (PM), frequency modulation (FM), phase shift keying (PSK), staggered quadrature phase shift keying (SQPSK). The RF driver portion typically multiplies up in frequency the internally or externally derived carrier to a frequency suitable for transmission.

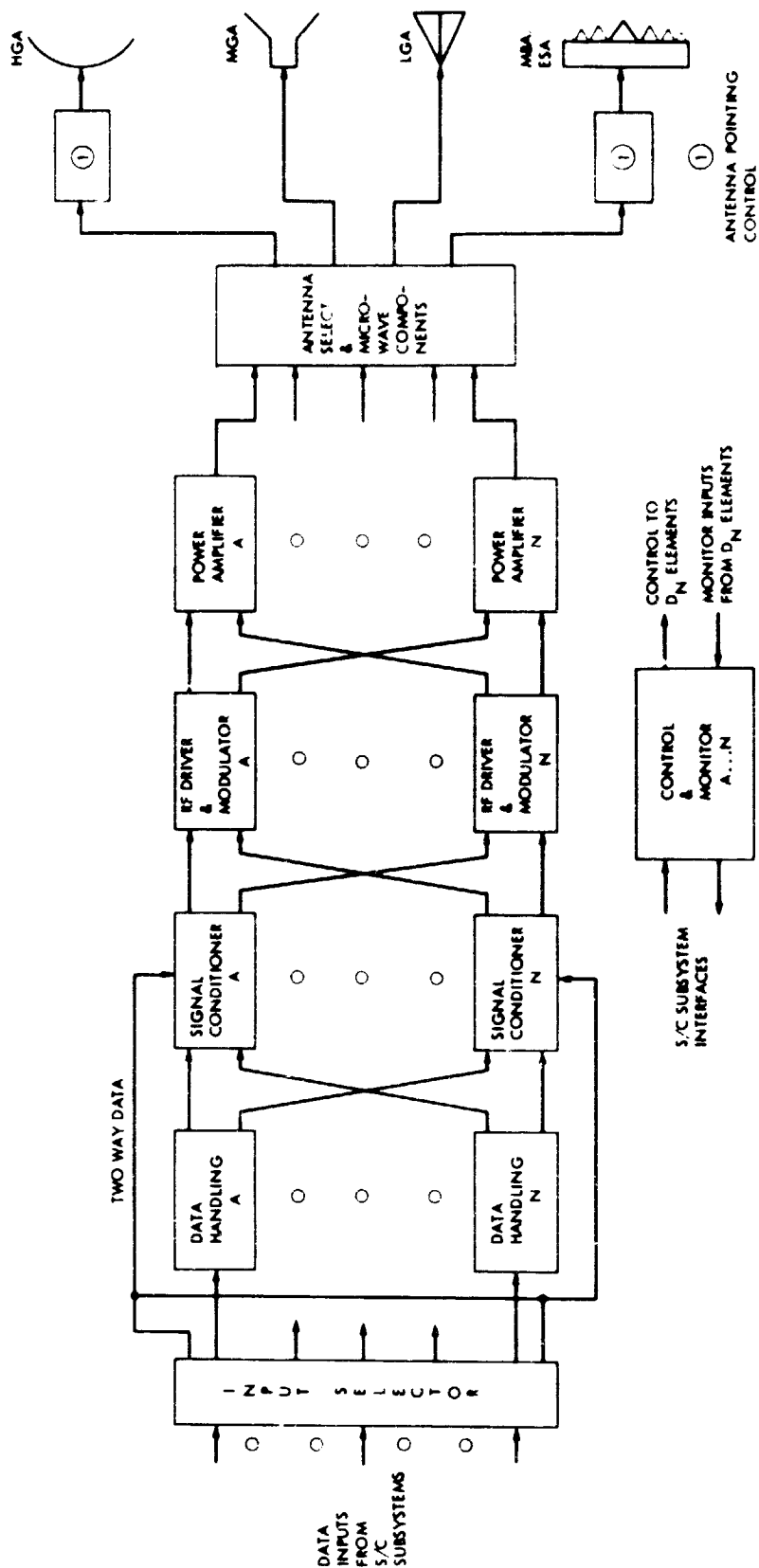


Figure III-15. Generalized TT&C Downlink (D_N) Functional Block Diagram

- (5) The Power Amplifier amplifies the RF carrier to a level suitable for transmission, e.g., Traveling Wave Tube Amplifier (TWT), Klystron, gallium arsenide field effect transistor (GaAs FET) Solid State Amplifier (SSA).
- (6) The antenna select elements provide for splitting, steering and filtering of the RF signal(s) to the antennas.
- (7) The antenna(s) direct the RF energy to the user. These antennas could have various gains and beamwidths, work at different frequencies and be used in various combinations depending on the mission.
- (8) With some of the antennas there could be a system for pointing the antennas electrically or electro-mechanically; others might be pointed only by satellite positioning by attitude control.
- (9) The control and monitor function for the downlink is essentially as described for the uplink in 3.1.1.1 (9) above.

3.1.1.3 Tracking Functional Elements. The tracking function processes a signal used to locate the satellite and determine its position and velocity state. One-way and two-way schemes for uplink and downlink tracking provide tracking station relative angular position and rate, range, and range rate in some combination dictated by mission requirements for navigation. The tracking functional block diagram is in Figure III-16.

The elements of the tracking function are:

- (1) A data input selector. This element receives various input signals such as a carrier and ranging signal from a receiver or multiple receivers.
- (2) The tracking and data conditioner receives the uplink data (e.g., carrier and/or ranging) for two-way tracking. This element could also be used to provide a stable frequency reference source for one-way tracking.
- (3) The output selector element provides the various tracking signals to the downlink transmitter(s).
- (4) The control and monitor element for the tracking function is essentially as described in 3.1.1.1 (9) above.

3.1.2 Autonomous Maintenance Functions

No required autonomous welfare maintenance functions were identified for the TT&C subsystem.

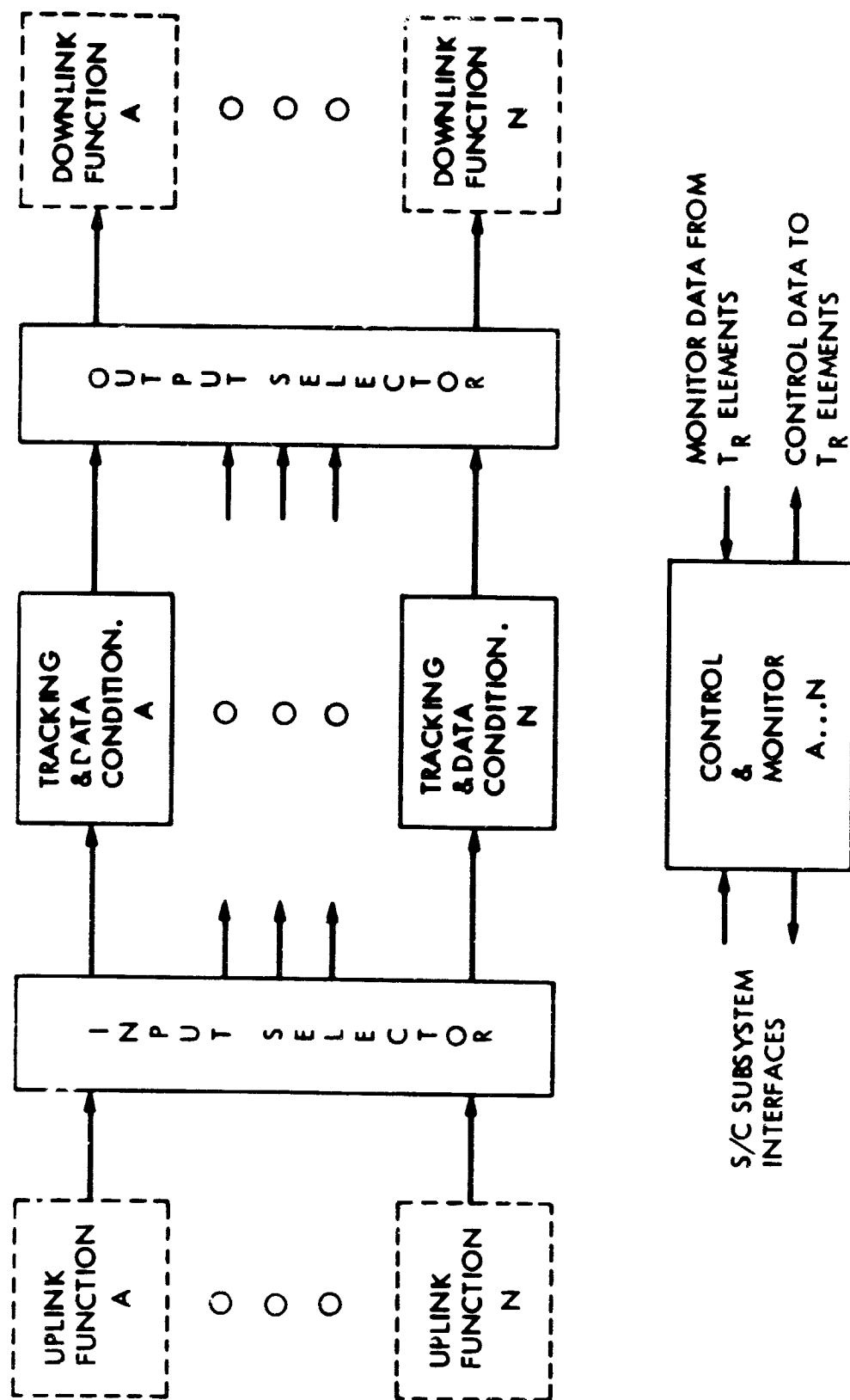


Figure III-16. Generalized TT&C Tracking (T_R) Functional Block Diagram

3.1.3 Autonomous Fault Management Function

Fault management functions for the three TT&C subsystem functional areas are characterized in Tables III-9, -10, and -11. The generic faults selected for characterization may be summarized as element hardware failures, antenna pointing control failures, and power failures.

Examples of flight-implemented algorithms for loss of uplink command capability (Viking spacecraft) and loss of downlink radio frequency capability (Voyager) are included in Appendix C.

Table III-9. Uplink TT&C Fault Management (Sheet 1 of 5)

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYMPOMS	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
<p>1.0 UPLINK FUNCTION (see Figure III-14)</p> <p>Provides for the reception of data on an uplink RF carrier. This data is typically for command and two-way tracking.</p>	<p>o TT&C Hardware failures</p>	<p>o No or bad command data.</p> <p>o Bad two-way tracking data (e.g., carrier and ranging).</p> <p>o Failure to issue command instructions to the subsystems or issuing a wrong command.</p>	<p>o Inability to control state of S/C from ground by commands.</p> <p>o Failure to issue preprogrammed commands.</p> <p>o Failure to store data transmitted on the uplink (e.g., modify on-board software).</p> <p>o A remote possibility (with good design) of issuing the wrong command.</p>	<p>o Direct test</p> <p>Inject test signals at the antenna or other functional element and test results.</p>	<p>o Test function directly.</p> <p>o Use Indirect sensors to localize fault.</p>	<p>o If isolated, switch to redundant unit.</p> <p>o If not isolated switch elements in sequence until fault is cleared.</p>	<p>o No known earth-elite with on-board direct test ability. It is complex hardware.</p>
				<p>o Indirect Test</p> <p>Examine fault sensors (e.g., telemetry) for out of tolerance condition.</p>	<p>o The sensor type and location could isolate the failed element.</p>	<p>o If the failed element is isolated switch to redundant unit. If not isolated, switch element in preset sequence until fault is cleared. Probably use cyclical switch of elements.</p>	<p>o Voyager used lack of command event to start switching elements.</p> <p>o Viking monitored telemetry data in post mission operations.</p>
				<p>o Active Redundancy</p> <p>Two receiving systems active. The command detector judges the quality of the command inputs (there could be two or more). If one input is bad, it uses the other.</p>	<p>o Isolation accomplished as part of detection. Detection only indicates one receiver string is bad and not necessarily a particular unit. Cross-strapping of elements is of questionable value in this approach.</p>	<p>o Correction is accomplished by the detector using the path with the best looking or most correct command data.</p>	<p>o Many earth elites use this approach: ISEE, LANDSAT, SMM. DSCS III uses a dual-frequency uplink approach.</p>

Table III-9. Uplink TT&C Fault Management (Sheet 2 of 5)

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYMPOMS	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
	o Bad antenna pointing due to antenna pointing system or S/C ACS pointing.	o As above.	o As above.	<ul style="list-style-type: none"> o Direct test of antenna pointing would be extremely complex. Indirect measurements could be used to judge faults. <p>NOTE: With S/C station keeping, antenna pointing could be monitored for faults in conjunction with ACS position knowledge.</p> <ul style="list-style-type: none"> o For fixed pointed antennas S/C attitude problems would be detected by ACS 	<ul style="list-style-type: none"> o Indirect measurements could indicate a particular antenna problem. o ACS sensors are used to isolate S/C attitude problems. 	<ul style="list-style-type: none"> o Problems might be corrected by switching to alternate antennas. In particular to lower gain, broad coverage antennas until the problem is corrected. o ACS provides fault protection for S/C attitude. 	<ul style="list-style-type: none"> o Voyager will switch to a low gain antenna automatically. o Many satellites have simple to complex designs for regaining S/C attitude stabilization.
	o S/C power failure	o As above.	o As above.	<ul style="list-style-type: none"> o Done by Power but would be detected as hardware failures by TT&C sensors. 	<ul style="list-style-type: none"> o If there was no input power the fault would be assumed to be in the power source (or a fuse or wire or connector). 	<ul style="list-style-type: none"> o Correction would be by the power s/e. Note that power would almost never be turned off on purpose to the command functions. 	<ul style="list-style-type: none"> o Viking, Voyager, DSCS III, and most of the complex S/C have power loss saving routines.

Table III-9. Uplink TT&C Fault Management (Sheet 3 of 6)

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYMPOMS	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
	o Hostile jamming	o As above.	o As above	<ul style="list-style-type: none"> o With a direct function tester the channel would check bad. o Most indirect measurements would not indicate a problem 	<ul style="list-style-type: none"> o Probably be guessed to be a hardware failure by the S/C if indicated by a direct sensor. o Indirect sensors would not generally isolate a failure. 	<ul style="list-style-type: none"> o The S/C probably could not correct for jamming but would probably switch elements anyway. 	o Extensive anti-jamming features in a design would be mission dependent features.
1.1 Antennas Provides for the reception of the uplink signals.	o Opens and shorts or mechanical failures.	o As above.	o As above.	<ul style="list-style-type: none"> o A direct tester which injects an RF signal into the antenna might detect an open or short. o Indirect sensors (e.g., reflected and incident power detectors) could detect faults. 	o Isolation is based on which antenna failed.	o Correction could be a switch of antennas.	o DSCS III X-Band could be switched from a medium gain antenna to a high gain antenna.
1.2 Antenna Pointing Hardware Equipment to direct the antenna for reception of a signal from some point in space (typically from the earth). Works with S/C attitude system.	o Component failures, electrical mechanical interface failures, mechanical failures.	o As above	o As above	<ul style="list-style-type: none"> o See 1.0 "Bad Antenna Pointing" above. o Direct test-complex but possible with S/C station keeping. o Indirect monitors <p>Continued.....</p>	<ul style="list-style-type: none"> o See 1.0 above. o Isolation to pointing electronics (typically part of CAM), mechanical or electrical pointing mechanism by indirect sensors (eg, shaft encoder position indication vs cmd. state). 	<ul style="list-style-type: none"> o See 1.0 above. o Correction would typically be a switch to another antenna. Block redundant antennas are not typical to a de-sign-but are possible. 	<ul style="list-style-type: none"> o See 1.0 above. o DSCS III uses mechanical and electrical beam steering as well as S/C "attitude" steering.

Table III-9. Uplink TT&C Fault Management (Sheet 4 of 6)

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYMPTOM	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
				<ul style="list-style-type: none"> For "autotrack" antennas (e.g., monopulse & con-scan) uplink signal is required. Indirect sensors could be used for fault detection. 			
1.3 Antenna Select and Microwave Components Directs and filters the RF signal from the antenna to the receiver input.	<ul style="list-style-type: none"> Electrical failures and mechanical failures. 	<ul style="list-style-type: none"> As above. 	<ul style="list-style-type: none"> As above. 	<ul style="list-style-type: none"> Direct tests difficult (except at total uplink level). Indirect sensors such as status or switch position indicators, incident and reflected RF power meas. 	<ul style="list-style-type: none"> Isolation by in-direct sensor purpose/location. 	<ul style="list-style-type: none"> Selection of alternative RF paths. 	<ul style="list-style-type: none"> Switch or component failures might result in use of an alternate antenna path, not necessarily with the same capability.
1.4 RF Head and Down Converter Receive, filter and frequency reduction of signal. 1.5 IF, Tracking, Data Demodulating Further frequency reduction, amplification and demodulation of data. 1.6 Data Detection Detect the data in the demodulated signal.	<ul style="list-style-type: none"> Electrical and mechanical failures which degrade or fail to function. 	<ul style="list-style-type: none"> As above. 	<ul style="list-style-type: none"> As above. 	<ul style="list-style-type: none"> Direct tests are possible but complex. Indirect sensors (e.g., telemetry function like voltages, currents, AGC, SPG, detector lock status...). 	<ul style="list-style-type: none"> Direct sensor typically at "Uplink" level only. Isolation by sensor type and location (e.g., high current in Data Detection Hardware). 	<ul style="list-style-type: none"> Correction could be by switching entire "strings" of redundant hardware (e.g., from A to B). If failure is isolated, a redundant element can be switched. 	<ul style="list-style-type: none"> Future technology may allow more "economical" implementation of direct and indirect sensors at this hardware or element level.

Table III-9. Uplink TT&C Fault Management (Sheet 5 of 6)

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYMPOMS	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
1.7 <u>Data Handling</u> Provides for data decoding, storage, preprogrammed commands, issuance of commands to the S/C (via the output selector). This is the command processing function based on direct uplink command inputs, stored and preprogrammed commands. Also, on-board autonomy related events would be processed via this element.	o Electrical failures. If bulk data storage is used possibly tape or electro/mechanical failures.	o As above. Usually will result in no command instruction issued or less frequently result in a bad command event.	o As above.	o Direct tests of data handling could be implemented with moderate complexity. Could be checked with total uplink direct test. (e.g., periodic injection of various data patterns and information with input and output checks). o Indirect: sensors could be used (e.g., command execution status, voltages, currents, decryption status).	o Fault isolation will depend on the form of direct or indirect sensing required by the mission.	o Correction typically would be a switch to the redundant element. In the case of a complex data handling system there may be cross strapping between redundant units (e.g., bulk non-volatile memory such as a tape recorder would probably be used in some redundant form). Systems with flexible autonomy features would probably need data storage on the uplink.	o The data handling for the uplink is very mission dependent. Depends on frequency of commanding, on board command updates, amount of data storage, amount of control required for autonomy.
1.8 <u>Data Output/Selector</u> o Directs and buffers the command instruction output to the appropriate subsystem.	o Typical electrical or mechanical failures.	o Failure to issue a command or issue a bad command.	o As above.	o Direct sensor test. o Indirect sensor test typically event status at output.	o Generally isolated by event status and the sensor types and locations.	o If redundant, the redundant unit might be selected. If internally redundant no action may be required.	o Sometimes designed with internal redundancy (e.g., quad-relays).

Table III-9. Uplink TT&C Fault Management (Sheet 6 of 6)

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYNTOMS	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
1.9 Control and Monitor (C&M) Provides control to the uplink functional elements (eg, switch redundant units). Primary interface with other S/C subsystems including inter-faces required for autonomy. Provides, for example, telemetry and status data for telemetry and accepts command inputs or data transfer inputs from the uplink command function.	o Typical electrical faults.	o Bad output sensor data (e.g., telemetry of fault sensors). o No reaction to command inputs or the wrong action. o Loss of autonomy features. o Failure to change modes or states. o Failure to switch redundant units.	o Inability to control or monitor some or all of the TT&C uplink elements.	o Direct testing possible but complex. o Indirect monitor of status or events or analog function could be used. o An autonomous system could observe the request for a TT&C state change and then monitor and C&M for a response. No response or an improper response would be used to indicate a C&M fault.	o Isolation by sensor function and location.	o Correction may be built into the design by "automatic" redundancy like quad power control relays. o Redundant unit could be selected or redundant elements within the C&M selected. o This unit would have most of the "autonomy" features included and might require re-programming if a new processor or distributed processing unit is selected.	This C&M element may not need to be redundant or fail safe in all designs (e.g., with a no-single-point-failure requirement, this "box" can fail to switch a redundant unit because it is the "second" failure).

Table III-10. Downlink TT&C Fault Management (Sheet 1 of 5)

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYMPTOMS	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
<p>2.0 Downlink Function</p> <p>(See Figure III-15)</p> <p>This function provides for the transmission of information over a downlink signal. Typically used for telemetry data, and two-way tracking data.</p>	<p>o TT&C Downlink hardware failures.</p>	<p>o No downlink signal.</p> <p>o No data on the downlink.</p> <p>o Weak signal.</p> <p>o Bad data.</p> <p>o Noisy or degraded data and/or carrier.</p>	<p>o No or bad telemetry data.</p> <p>o No or bad carrier tracking data.</p> <p>o S/C status unknown.</p>	<p>o Direct downlink test-sample antenna RF output signal and test results (i.e., receive, demodulate detect and decode data). - Could be done at functional element level or downlink system level.</p> <p>o Indirect sensors. Typically telemetry analog & digital signals plus other specialized sensors as needed or required for fault detection.</p> <p>o Active redundancy requires no detection. Redundant units are powered simultaneously. Typically different frequencies would be used.</p>	<p>o Direct test might only isolate problem to a particular set of elements.</p> <p>o Indirect sensor location and type.</p>	<p>o Switch to redundant unit if in passive standby.</p> <p>o Switch elements in sequence if fault location not isolated.</p> <p>o For multiple failures cross-strapping might be utilized in more complex systems.</p>	<p>o With two completely active downlink systems. Switching would not be required for single failures.</p>
	<p>o Bad antenna pointing due to antenna pointing system or S/C ACS pointing.</p>	<p>o As above.</p>	<p>o As above.</p>	<p>o Direct antenna pointing test would be very complex-with S/C station keeping is easier.</p> <p>Continued. . .</p>	<p>o Isolation by sensor type and location.</p> <p>o ACS would isolate S/C attitude faults.</p>	<p>o Select alternate antenna (e.g., lower gain broad coverage antenna).</p> <p>Continued. . .</p>	<p>Antennas and antenna pointing typically would not be block redundant. Pointing electronics might be redundant.</p>

Table III-10. Downlink TT&C Fault Management (Sheet 2 of 5)

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYMPTOMS	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
<p>o Typical electrical equipment failures.</p>	o S/C Power failure.	o As above.	o As above.	<p>o For S/C attitude faults-ACS would provide detection.</p> <p>o Indirect sensors could be used to indicate hardware faults.</p> <p>o Detection by the Power Subsystem. However, direct and indirect TT&C sensors would indicate a massive failure.</p>	<p>o Sensors output would indicate a probable input power failure; (however, a fuse, wire or connector failure probably would not be isolated as the failure without very specialized multiple sensors).</p>	<p>o Switch to redundant antenna pointing electronics, if available.</p> <p>o ACS would regain S/C stabilization.</p> <p>o Correction would be accomplished by the power subsystem.</p> <p>o S/C power down states (i.e., S/C safing procedures) probably would turn off the transmitters. These procedures should have power up procedures also.</p>	<p>o The transmitters are large power drains. With a S/C power problem they would typically be turned off, then back on when the problem was cleared.</p>
	o Typical electrical equipment failures.	o As above. But, typically, selected measurements would be bad.	o As above.	<p>o Direct test of all of the inputs of this function would be very complex (e.g., simulate 1000 or so analog & digital signals). However, many elements could be checked indirectly. telemetry monitors would provide limited fault detection.</p>	<p>o Isolation of failures difficult but possible.</p> <p>o Using redundant inputs & switching eliminates the need for isolation.</p> <p>o Indirect sensors could indicate a bad "active" unit of a redundant pair.</p>	<p>o No change if input/output redundancy built in.</p> <p>o Switch to redundant unit if available.</p>	<p>o These input/output buffers could be implemented with built-in redundancy or designed to be insensitive to some failure modes.</p>

Table III-10. Downlink TT&C Fault Management (Sheet 3 of 5)

Fault Category	Generic Faults	Example of Fault/Symptoms	Impact of Fault	Fault Detection	Fault Isolation	Fault Correction	Example/Comment
Data Handling	Typical electrical hardware failures.	o As above. Specific problems might be unrecognizable data, non-coded data, no data, no stored data playback.	o As above.	o A direct function tester would be complex but could check many elements. o Indirect sensor types are limited in a practical sense but will indicate some failures. o More complex data systems could be "fault tolerant" with internal "error" detection and correction.	o Isolation might be to this whole element or possibly to an internal element (e.g., a magnetic bubble memory failure).	o Correction could be to switch to the redundant element or switch internal redundant units. Also, if a fault tolerant, self testing design, correction, could be made internally.	o This data handling element would probably have internal redundancy (e.g., two tape recorders, two multiplexers). The design of the data handling system is very mission dependent.
Signal (modem)	Typical electrical hardware failures.	o As above.	o As above.	o A direct tester could detect most failures of this element. o Indirect sensors could also detect some failures.	o Isolation by sensor type and location.	o Correction would typically be a switch to the redundant unit. o Correction could be to have alternate data modes (e.g., no sub-carrier, suppressed carrier, no encryption).	

Table III-10. Downlink TT&C Fault Management (Sheet 4 of 5)

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYMPOMS	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
2.4 RF Driver and Modulator Modulates the data onto the carrier, multiplies up the frequency and amplifies the signal.	o Typical electrical failures.	o As above. Could be loss of modulation, loss of total signal, or a poor quality signal.	o As above.	o A direct downlink tester could be used. o Indirect sensors could be used.	o Isolation by sensor location and type.	o Correction would typically be selection of redundant unit.	
2.5 Power Amplifier Amplifies the RF signal to the level required.	o Typical electrical failures.	o As above. Typically a loss or reduction of radiated RF power.	o As above.	o A direct downlink tester could be used. o Indirect sensors could be used (e.g., RF output power).	o Isolation by sensor type and location.	o Correction would typically be a switch to a redundant unit.	o Most Satellite Systems monitor the total RF power output via Telemetry. o Mariner, Viking and Voyager autonomously switched to the redundant unit if output power was below a prescribed level. o DSCS III used active redundancy by having TWT X-band amplifiers on continuously, at different frequencies.
2.6 Antenna Select and Microwave Components These components filter the output and direct it to the antenna(s).	o Some electrical failures, but mechanical type failures dominates (e.g., connector pin retraction).	o As above. Typically loss of or reduction of RF power. Also, RF power radiated from the wrong antenna.	o As above.	o Direct test difficult except at total downlink level. o Indirect sensors limited (e.g., incident and reflected power, switch position status).	o Isolation by sensor type and location no RF from HGA).	o Typically non-redundant. Alternate paths would have to be used if available (e.g., select a low gain antenna path if the HGA path fails).	o These elements frequently represent "single point" failures in an antenna path.

Table III-10. Downlink TT&C Fault Management (Sheet 5 of 5)

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYMPOMS	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
2.7 Antenna Pointing Hardware Equipment to direct the antenna radiated RF signal at some point in space (typically toward some point on earth). Has to work with the S/C attitude control system.	o Typical electrical failures in the pointing electronics. More unique failures in the electromechanical hardware, or in the "electrically" steerable hardware.	o As above. Typically if the antenna is pointed wrong there is a significant reduction in downlink level.	o As above.	o See 2.0 "Bad antenna pointing." above. o If the S/C has station keeping, ACS and indirect fault sensors might detect bad antenna pointing. o Direct testing of RF pointing is very complex.	o Isolation by indirect sensor type and location (e.g., shaft encoder or resolver). o Some of the "steering" equipment (e.g., motor, flexible RF joints) may not be redundant. o Alternate lower gain antennas may have to be selected. o S/C attitude could be used to point antenna.	o Electronics could be redundant and switched. o Some of the "steering" equipment (e.g., motor, flexible RF joints) may not be redundant. o Alternate lower gain antennas may have to be selected. o S/C attitude could be used to point antenna.	o DSCS III X-Band uses an electrically steerable multi-beam and a "mechanically" steered gimbaled dish, (i.e., HGA).
2.8 Antennas Radiate the RF energy to a location in space with wide to narrow radiation patterns (typically pointed at the earth).	o Typically opens and shorts, of a mechanical nature (e.g., connector failure).	o As above. Typically a reduction in the radiated signal level.	o As above.	o A direct tester could be used by sampling the radiated RF signal. o Indirect sensors could be used but are limited (e.g., incident & reflected power).	o Isolated by sensor type and locations. Also, typically only one antenna is radiating.	o Correction by switching to alternate antennas possibly with changes in downlink signal strength.	o Antennas are seldom block redundant but are frequently functionally redundant (e.g., medium gain and a high gain antenna).
2.9 Control and Monitor (C&M) Provides control to the various downlink elements (e.g., redundancy switching, antenna pointing, mode switching). Provides monitoring of downlink functions. Can be used for telemetry outputs; data to other subsystems, sensor data for autonomy. Primary interface with other subsystems.	o Typical electrical faults.	o Some of the above. o Failure to switch a redundant element. o Failure to change states or modes. o Bad monitor outputs for telemetry to other subsystems or for autonomy.	o Inability to control the downlink functions. Could result in the impacts above.	o Direct testing possible, but complex. Fault tolerance and self checking (direct test) design is possible. o Some indirect sensors can be used. Mostly these would be status or event sensors (e.g., if a command were sent and the state was not changed, a fault would be detected). Testing of the analog functions would be more difficult.	o Isolation by sensor type and location.	o Redundant unit could be switched in. o Built in fault tolerance such as quad relays would not require direct action.	o This unit may not need to be redundant or fail safe in all system designs. o Autonomy functions would be included in this unit.

Table III-11. Tracking TT&C Fault Management

FUNCTIONAL DESCRIPTION	GENERIC FAULTS	EXAMPLE OF FAULT/SYMPTOMS	IMPACT OF FAULT	FAULT DETECTION	FAULT ISOLATION	FAULT CORRECTION	EXAMPLE/COMMENT
<p>3.0 TRACKING FUNCTION (see Figure III-16)</p> <p>The tracking function provides turned around uplink signals on the downlink for two way tracking and a downlink signal for one way tracking. These signals are used by the ground for satellite position determination. The uplink and downlink functions are required for tracking.</p>	<ul style="list-style-type: none"> o The uplink and downlink type faults will fail the tracking functions. In addition, the additional tracking hardware also can have typical electrical faults. 	<ul style="list-style-type: none"> o Loss of the downlink carrier and/or ranging data; loss of phase or frequency coherence, noisy carrier; no ranging data. 	<ul style="list-style-type: none"> o Tracking data not available or bad. Could result in improper spacecraft location determination, or lack of position determining ability. 	<ul style="list-style-type: none"> o Fault detection techniques used for uplink and downlink functions apply here. o Direct testing of this function is complex. o Indirect sensors can be used to detect some types of failures. 	<ul style="list-style-type: none"> o Fault isolation techniques used for uplink and downlink functions apply. o Fault isolation by sensor type and location. 	<ul style="list-style-type: none"> o Fault correction techniques for the uplink and downlink functions apply. o Fault correction by switching to redundant unit, or alternate path or by capability reduction (e.g., no ranging). 	<ul style="list-style-type: none"> o The use of the uplink signal and the downlink signal for tracking requires most of the functional elements of the up and downlink. In addition, an interface box is required between the up and downlink to transfer or translate the data.

3.2* POWER SUBSYSTEM

3.2.1 Functional Description and Elements

The Power subsystem is responsible for the generation, conditioning, distribution, and management of electrical power for the spacecraft. Autonomous design characteristics affect many elements of the subsystem. A representative set of subsystem elements is:

3.2.1.1 Batteries. Secondary (rechargeable) energy storage batteries require several control functions to provide optimum performance. Depth of discharge control is an important function because of the direct relationship between decreasing battery cycle life and increasing depth of discharge. Depth of discharge is usually determined by calculating the time integral of discharge current (Ampere-Hours) and relating this value to the rated battery capacity (Ampere-Hours). Temperature control of batteries is used to achieve maximum energy conversion during both charge and discharge phases. Periodic reconditioning of Nickel-cadmium (NiCd) batteries has been shown to result in the recovery of energy storage capability which degrades with life and charge-discharge cycling.

3.2.1.2 Battery Chargers. Battery charger control is necessary to prevent excessive overcharge which generates heat in the battery and causes degradation. In addition to battery over-temperature and maximum voltage cutoff, charger control may include charge rate at one or more selectable profiles based on specific battery voltage-temperature characteristics. Charger on-off is also incorporated to allow isolation of a failed charger.

3.2.1.3 Voltage Regulators. Voltage regulators, series or shunt type, are usually self-contained functional elements. Shunt regulators contain sufficient redundancy to achieve required fault tolerance. Fault tolerance for series regulators is accomplished by incorporating a spare redundant regulator unit. Regulator control is therefore limited to switching to a redundant series regulator in the event of a fault.

3.2.1.4 DC-DC Converters. DC-DC converters are usually self-contained functional elements with no provisions for external control. Functional fault tolerance is controlled by standby redundancy switching.

3.2.1.5 DC-AC Inverters. Inverters are also usually self-contained functional elements without provision for external control. An exception to this occurs where synchronization of the inverter frequency to an external reference is required. Control of the inverter frequency in the event of loss is accomplished by automatic enablement of an internal reference. Operation would continue in an unsynchronized mode. Fault tolerance of the inverter is controlled by standby redundancy switching.

*By A. O. Bridgeforth

3.2.1.6 Load Power Switching. Load On/Off switching is usually performed within the power subsystem in response to inputs from the spacecraft command subsystem. Load fault control can be accomplished by fusing, current limiting or by activation of the On/Off switch.

3.2.1.7 Memory-Keep-Alive Power Supply. This type of power supply is used to supply the uninterrupted voltage required to maintain the state of "volatile" memory during power subsystem fault or transient conditions. Active internal redundancy is required, therefore no external controls are necessary.

3.2.1.8 Ordnance Power Switching Unit. Ordnance power switching is usually accomplished by a parallel redundant set of safe-arm and fire relays actuated in response to spacecraft command subsystem inputs.

3.2.2 Autonomous Maintenance Functions

Typical welfare maintenance functions were categorized into the areas of energy storage maintenance, subsystem performance assurance, load management, and margin determination. Maintenance functions falling in these categories are characterized in Table III-12.

3.2.3 Autonomous Fault Maintenance

Table III-13 identifies and characterizes a series of faults that may occur in elements of the Power subsystem. Examples of generic algorithms to implement a power load fault management function, a failed battery cell replacement function, and a failed DC-DC converter recovery are included in Appendix B.

Table III-12. Autonomous Maintenance Functions (Sheet 1 of 3)

Generic Function	Example of Maintenance	Scope Mission System Subsystem Impact	Subsystem Assys. Involved	Frequency of Execution	Inputs	Processing	Outputs
Energy Storage Management	Battery Recharging	Maintain sufficient energy to provide spacecraft power for all eclipse and load switching transient Provide sufficient energy to clear faults and allow restoration of safe spacecraft operation	Batteries Battery Chargers Solar Array	After each battery discharge event or Battery State of Charge - 95	<ul style="list-style-type: none"> • Battery Temp. • Battery State of Charge Indicator 	<ul style="list-style-type: none"> • Battery state-of-charge - 95 • Determine that battery charge power is available • Determine battery temperature and select appropriate voltage-temperature charge limit curve • Switch battery charger "On" • Verify battery charging by presence of charge current • Switch charger to "trickle charge" at selected voltage-temperature point • Reset state-of-charge indicator to 100 	<ul style="list-style-type: none"> • Battery state-of-charge indicator reads 100 • Battery charger status-trickle charge
	Battery Reconditioning	Maintain maximum battery capacity. NiCd batteries exhibit increasingly degraded capacity during cyclic partial discharge-charge operation. Periodic reconditioning can restore the original battery capacity	Battery Battery charger Battery Reconditioning Module	Prior to each eclipse season	<ul style="list-style-type: none"> • (x) days - start of eclipse period = timer signal. Where x depends on number of batteries and duration of reconditioning sequence for each battery • Battery temp. • Battery voltage 	<ul style="list-style-type: none"> • "Initiate reconditioning signal" from spacecraft timer, starts sequence • Verify solar array margin - 50 Watts • Switch "Off" battery charger • Switch "On" battery reconditioning load • Remove reconditioning load when battery voltage is reduced to 1 Volt • Initiate battery charge sequence 	<ul style="list-style-type: none"> • Battery state-of-charge indicator reads 100 • Battery charger status-trickle charge

Table III-12. Autonomous Maintenance Functions (Sheet 2 of 3)

Generic Function	Example of Maintenance	Scope: Mission/System/ Subsystem Impact	Subsystem Ass'y. Involved	Frequency of Execution	Inputs	Processing	Outputs
Power Margin Determination	Periodically determine, and update stored value of solar array excess power	Prevent unplanned battery discharge. Provide data on solar array degradation. Verify capability of solar array to support planned changes to load power profile	Solar Array Shunt Regulator Power Distribution Housekeeping functions	Once per day	Shunt Current Bus Voltage Total Solar Array Load Current	<ul style="list-style-type: none"> Calculate total solar array load power (Spacecraft loads plus PPS housekeeping) List all spacecraft and housekeeping loads in the "On" state Determine from shunt regulator current and bus voltage the source power margin 	<ul style="list-style-type: none"> Current load status Source power margin
Power Subsystem Performance	Subassembly status determination	Periodic determination of the performance status of each operating functional sub-assembly, and to detect potential failures from degradation data	Battery Charger DC-DC Converter DC-AC Inverter	Once each hour	<ul style="list-style-type: none"> Input voltage and current for each subassembly Output voltage and current for each subassembly Efficiency tables vs. input power for each subassembly 	<ul style="list-style-type: none"> Calculate the efficiency of each sub-assembly Compare each efficiency with the corresponding stored value at the designated input power level Update previous stored results Flag and/or execute appropriate alarms and corrective actions 	<ul style="list-style-type: none"> Updated status on all operating subassemblies Degradation data
Power Subsystem Performance	Self test of redundant functions	Periodically verify the performance capability of redundant off-line functions. Provide the capability to perform failure analyses on suspected failed units which were switched off-line	Battery Chargers DC-DC Converters DC-AC Inverters	Once per month or as required	Input voltage and current. Output voltages and currents	<ul style="list-style-type: none"> Switch the off-line unit "On" to a current limited source and test load Verify output voltage(s) are within tolerance Calculate efficiencies and verify within tolerance Execute appropriate flags and store data 	<ul style="list-style-type: none"> Updated status on all redundant off-line units Data to support analysis of anomalous unit

Table III-12. Autonomous Maintenance Functions (Sheet 3 of 3)

Generic Function	Example of Maintenance	Scope: Mission System Subsystem Impact	Subsystem Ass'y. Involved	Frequency of Execution	Inputs	Processing	Outputs
Load Management	Individual load monitoring	Determine the operating power level of each load to indicate normal/abnormal operation	Power Distribution & Switching	Once each hour	Voltage and current measurements of each load. On/Off status of each load switch Mode state of each multi-mode load Look-up table of power levels vs. mode state	<ul style="list-style-type: none"> Calculate the load power for each load Verify zero power level for each load in the "Off" state Verify power levels within tolerance for each non-switched and "On" loads for the given operating mode Execute appropriate flags/corrective actions 	<ul style="list-style-type: none"> Updated status verification of each load Supporting data for load anomaly or degradation analyses

Table III-13. Autonomous Fault Management Functions (Sheet 1 of 2)

Assembly/ Subassembly Functional Description	Generic Fault	Example of Fault/ Symptom	System/Subsystem Impact and Criticality	Detection/Sensing	Isolation	Recovery	Mission Examples
Solar Array Power Slip Rings	Failed Array Section(s)	Failed-oper Slip Rings	<ul style="list-style-type: none"> Reduced spacecraft performance capability Reduced battery and mission life 	<ul style="list-style-type: none"> Unplanned battery discharge Shunt current abnormally low 	N/A	<ul style="list-style-type: none"> Reduce power system loads 	SEASAT Failure
	Shorted Power Bus	Residue buildup between slip rings	<ul style="list-style-type: none"> Failure of the power system and mission (assuming insufficient energy to clear fault) 	<ul style="list-style-type: none"> Short circuit battery discharge current readings Falling bus voltage Increasing battery temperature Increasing temperatures at the fault site 	<p>Not possible unless isolation diodes are placed on the spacecraft bus side of the slip rings, which would prevent the battery and other array sections from powering the shorted slip ring</p>	<p>Same comment as under "Isolation"</p>	
Battery Charger	Abnormal Input and/or Output Current	No output current	<ul style="list-style-type: none"> No immediate impact Potential reduction in mission life 	<ul style="list-style-type: none"> Abnormal battery charge current Abnormal battery charger input current 	<p>Disconnect battery charger input and output</p>	<ul style="list-style-type: none"> Initiate program to cross strap operating battery charger after full charge of its normally connected battery Operate failed charger in a test configuration to obtain data for failure analysis 	
Battery	Reduced capacity	Degraded cell(s)	<ul style="list-style-type: none"> None immediately Potential reduction in mission life 	<ul style="list-style-type: none"> Abnormally low battery voltage Failure to charge to V-T limit 	Switch battery off-line	<ul style="list-style-type: none"> Initiate battery reconditioning sequence 	

Table III-13. Autonomous Fault Management Functions (Sheet 2 of 2)

Assembly/ Subsystem/ Functional Description	Generic Fault	Example of Fault/ Symptom	System/Subsystem Impact and Criticality	Detection/Sensing	Isolation	Recovery	Mission Examples
DC-DC Converter	Abnormal Output	Failed switching transistor	Immediate loss of con- verter load functions. Load function interrup- ted until redundant converter is switched on-line and load recovers.	<ul style="list-style-type: none"> Loss of load function Abnormally high or low converter input current Abnormally low conver- ter output voltage(s) 	Disconnect con- verter inputs and outputs	<ul style="list-style-type: none"> Switch to redundant converter Go to failed converter to a test load Measure and store appro- priate data for analysis by the ground segment Set appropriate status flag, i.e.: converter operational, failed, emergency use 	MM 73
Power Distribu- tion	Load Fault	High Voltage Arc in the load	Total instrument failure or loss due to blown fuse	Change in load resistance on the power bus	Load disconnect	<ul style="list-style-type: none"> Periodically compute the resistance of each load Compare the computed value with limit values stored in memory If the value is not within limits, switch the load "off" and set the status flag Reconnect of the load can be safely accomplished to determine if the fault was due to a transient occurrence If the load has degraded but not failed, the stored limit values can be changed and the load reconnected 	

3.3* ATTITUDE CONTROL SUBSYSTEM

3.3.1 Functional Description and Elements

A three-axis stabilized spacecraft attitude control subsystem (ACS) typically consists of point source and/or extended body sensors for attitude references, a set of actuators (momentum wheels, thrusters, etc.) to provide control response, data or signal bus structures for internal information flow, and a central control logic authority. The following are typical elements of these functional subdivisions with implications for autonomous ACS design.

3.3.1.1 Reaction Wheels. Reaction wheel assemblies require control for momentum unloading, momentum distribution management (multiple active wheel assemblies), and fault recovery.

3.3.1.2 Sensors. Attitude reference sensors may utilize near bodies (sun, moon, earth), stellar references, or vehicle body dynamics (gyroscopes). Control must typically be exerted to calibrate sensors, initialize redundant sensor assemblies (block or functional), and switch subsystem operating modes for error recovery.

3.3.1.3 Thrusters. Attitude control thrusters may be considered part of the propulsion subsystem under control of the ACS. Thruster selection and enabling, fuel management, and fault detection and response are typical control requirements. Responsibility for control of specific functions may lie with the ACS, propulsion subsystem, or a spacecraft system executive.

3.3.1.4 Computer/Attitude Control Electronics. Programmable logic resources for closed loop control of ACS functions may reside in a general purpose computer or a specialized logical control subassembly. Such devices must be commanded to switch logical operating modes for performing attitude control in changing spacecraft operating modes (reference acquisition, maneuver execution, normal operations, etc.) and to respond to a wide variety of internal and external faults that may impact subsystem operation.

3.3.2 Autonomous Maintenance Functions

Attitude control maintenance functions are grouped into areas of momentum management, operating mode sequencing and configuration, audit trail maintenance, and evaluation of ACS subassembly performance. These characterizations are presented in Table III-14.

By J. Matijevic

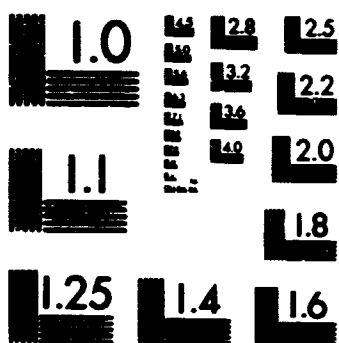
Table III-14. ACS Autonomous Maintenance Functions (Sheet 1 of 7)

Generic Function	Example of Maintenance	Scope: Mission/System/ Subsystem Impact	Subsystem Assemblies Involved	Frequency of Execution	Inputs	Processing	Outputs
Attitude Momentum Management	Momentum Wheel Unloading	Spacecraft attitude is maintained and wheels saturated from spin up/ saturation Long periods of absence of ground visibility/ autonomous operation require maintenance outside of fault recovery activities	Reaction wheels Thrusters	Stored momentum ex- ceeds TBD% of max/mum or momentum unloaded to level needed for per- formance of maneuver, turn, re- acquisition of references, etc.	<ul style="list-style-type: none"> Tachometer readings; Momentum and attitude state vectors; active wheel and thruster configuration; time of unloading sequence/performance requirements 	<ul style="list-style-type: none"> Tachometer readings indicate need for unloading sequence; Sequence scheduled; Attitude and momentum states monitored/used as sequence proceeds; Tachometer readings verify momentum reduction; Tachometer readings or time elapsed indicates end of sequence 	<ul style="list-style-type: none"> Thruster firing commands New, estimated momentum updates
Mode sequencing/ configuration	Normal configuration	Autonomous fault recovery requires the sequencing from re-stabilizing modes to normal operations, and the establishment of a stable configuration	Sensors Actuators Processor/ memories Interface electronics	After the completion of re- acquisition of ref- erences, commanded maneuvers, or other non-normal operating mode or after special maintenance procedures have been completed in the normal operating mode	<ul style="list-style-type: none"> System executive enable Sensor states Potentiometer reading Delay timers for device powered states Disabled devices/configuration flags 	<ul style="list-style-type: none"> Enable allows sensor states or potentiometer readings to time sequence into normal configuration Set up/monitor powering up of any devices Switch to/enable use of healthy devices and electronics End sequence with nominal sensor or potentiometer readings 	<ul style="list-style-type: none"> Power up/configuration commands

83

69.20

3/B



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

Table III-14. ACS Autonomous Maintenance Functions (Sheet 2 of 7)

Generic Function	Example of Maintenance	Scope: Mission/System/ Subsystem Impact	Subsystem Assemblies Involved	Frequency of Execution	Inputs	Processing	Outputs
Mode sequencing/ configuration	Eclipse occultation	<p>In autonomous operation solar eclipses or stellar occultations must be distinguished from sensor faults which have caused loss of references</p> <p>If solar power is used, battery power must be available during solar eclipses</p>	<p>Sun sensors</p> <p>Star trackers</p>	<p>As predicted by an auto-navigation system or as available from ephemeris information transmitted from the ground</p>	<ul style="list-style-type: none"> • System executive enable • Time and duration of eclipse or occultation • Sensor readings and states • Momentum and attitude state vectors 	<ul style="list-style-type: none"> • Enable allows sensor readings to determine the beginning and end of the period of occultation or eclipse. (Use reduction in intensity) • Disable sensor fault protection and actuator fault protection which relies on sensor readings • Enable low torque thruster activity • Safe payload as appropriate • Switch to alternate sensor configuration to avoid loss of control as appropriate • Monitor attitudes during occultation • Reenable sensor fault protection and reconfigure as needed at the end of occultation 	<ul style="list-style-type: none"> • Sensor configuration commands • Attitude performance parameters

Table III-14. ACS Autonomous Maintenance Functions (Sheet 3 of 7)

Generic Function	Example of Maintenance	Scope: Mission/System/ Subsystem Impact	Subsystem Assemblies Involved	Frequency of Execution	Inputs	Processing	Outputs
Audit trail	Audit trail of fault protection	Ground visibility is needed for the analysis and validation of a subsystem's fault response. During periods of autonomous operation, such visibility may be pro- vided by an audit trail associated with the fault	Sensors Actuators Processor/ memories	During fault protection activity and periodically as requested by system executive	<ul style="list-style-type: none"> • Sensor readings • Potentiometer readings • Attitude and momentum state vectors • Actuation parameters • Time tags • Performance parameters • Fault response flags and states • System executive readout enable 	<ul style="list-style-type: none"> • Fault detection algorithm determines an alert limit has been exceeded • Sensor reading, state vectors, fault response flags etc. as appropriate are stored and time tagged for audit trail • If detection algorithm determines an alarm limit has been exceeded isolation and correction activities are enabled, and indication output to audit trail • Sensor readings state vectors, etc. output until sensor/actuator response below alert limits • At system executive enable, audit trail, locally stored, is output to system located non-volatile storage unit 	<ul style="list-style-type: none"> • Time tagged parameter of subsystem fault response

Table III-14. ACS Autonomous Maintenance Functions (Sheet 4 of 7)

Generic Function	Example of Maintenance	Scope: Mission/System/ Subsystem Impact	Subsystem Assemblies Involved	Frequency of Execution	Inputs	Processing	Outputs
Subassembly evaluation	Gyro drift	To perform autonomous rate damping, attitude control and sensor rate calibration with gyros, drift must periodically be deter- mined and compensated for	Rate integrating gyros, celestial sensors	After an inertial mode has been sequenced through and during a normal or cruise oper- ating mode	<ul style="list-style-type: none"> • System execu- tive enable • Integrated angle error signals from gyros • Celestial sensor error signals • Drift toler- ance limits 	<ul style="list-style-type: none"> • Enable allows gyros on during cruise or normal operating mode • Correlate error signals from gyros with those produced by cele- stial sensor over TBD time interval • Develop compen- sating drift param- eter from gyro signals outside of tolerance limits as compared to sensor signals • Store drift compen- sations for next/ subsequent use of gyros and turn gyros off 	<ul style="list-style-type: none"> • Gyro drift compensation

Table III-14. ACS Autonomous Maintenance Functions (Sheet 5 of 7)

Generic Function	Example of Maintenance	Scope: Mission/System/ Subsystem Impact	Subsystems Assemblies Involved	Frequency of Execution	Inputs	Processing	Outputs
Subassembly evaluation	Star tracker performance	To provide precision attitude control during extended periods of autonomous opera- tion, performance under control of an attitude sensor must be analyzed. Abnormal thruster duty cycles or use of fuel could be the result of noise in this sensor	Star tracker	TAD days as determined by modeled sensor char- acteristics and bypassed if fault re- covery or maneuver occurred during the period for analysis.	<ul style="list-style-type: none"> Star tracker readings Attitude and momentum state vectors Star threshold limits, magnetic field and temperature reading Time of the tracker readings Attitude biases Tachometer readings and momentum dumping parameters Thruster and fuel usage parameters 	<ul style="list-style-type: none"> Compare accumulated star tracker readings against stored performance parameters Or compare attitude or momentum values on axes of control against performance parameters And/or note amount of fuel usage, number of thruster firings and number of momentum dumping operations and compare with nominal performance Develop updates to tracker star threshold limits, or attitude bias values based on non-nominal performance Schedule switch in trackers if performance beyond stored alarm tolerances 	<ul style="list-style-type: none"> New attitude biases or star tracker threshold limits Request for swap in trackers

Table III-14. ACS Autonomous Maintenance Functions (Sheet 6 of 7)

Generic Function	Example of Maintenance	Scope: Mission/System/ Subsystem Impact	Subsystem Assemblies Involved	Frequency of Execution	Inputs	Processing	Outputs
Attitude/ Inertial Reference Calibration	Star-sensor boresight alignment and gain	Spacecraft attitude is degraded due to mechanical, or elec- trical drifts in the star sensor	Star sensor, attitude control electronics	Depends on spacecraft environment, required precision, and sensor properties	<ul style="list-style-type: none"> Star sensor signals 	<ul style="list-style-type: none"> Sensor shutter closed and bore-sight source turned on Sensor output compared with stored values Corrections to cali-brations computed 	<ul style="list-style-type: none"> New values of star tracker calibration constant
	Earth sensor drift correction	Earth pointing is de- graded due to detec- tor and amplifier drifts	Earth sensor, attitude control electronics	Depends on spacecraft environment, required precision, and sensor properties; calibration carried out during periods of limited payload operation	<ul style="list-style-type: none"> Earth sensor signals 	<ul style="list-style-type: none"> Spacecraft pointed away from earth, sun, or moon Sensor output recorded Calibration factors calculated Spacecraft returned to earth pointing 	<ul style="list-style-type: none"> New values of earth sensor calibration constants
	Gyro calibration	Inertial guidance, maneuver accuracy will be degraded due to incorrect gyro drifts, scale factors and biases	Gyros, star sensors, sun sensors, atti- tude control electronics	Depends on required precision and gyro properties calibration carried out during non- critical mission phases	<ul style="list-style-type: none"> Gyro readings Star sensor signals Sun sensor signals 	<ul style="list-style-type: none"> Spacecraft placed in a sequence of attitudes Gyro and star sensor outputs stores for each attitude Gyro calibration factor computed for each attitude 	<ul style="list-style-type: none"> New values of gyro calibration constants

Table III-14. ACS Autonomous Maintenance Functions (Sheet 7 of 7)

Generic Function	Example of Maintenance	Scope: Mission/System/ Subsystem Impact	Subsystem Assemblies Involved	Frequency of Execution	Inputs	Processing	Outputs
Attitude/ Inertial Reference Calibration (Cont)	Sun-sensor gain adjustment	Spacecraft attitude is degraded due to change of sensor sensitivity following a nuclear event	Sun sensor, nuclear-event detector attitude control electronics	Once after the detection of a nuclear event	<ul style="list-style-type: none"> Sun sensor signals Nuclear-event detector reading 	<ul style="list-style-type: none"> Nuclear-event detector triggered Sun-sensor output compared with pre-event value Estimation algorithm uses model of sensor behavior to adjust gain and restore sensor output 	<ul style="list-style-type: none"> New sun-sensor gain
	Attitude sensor inter-comparison	Redundant sensors are intercompared to detect inconsistencies. Uncorrected inconsistencies may lead to degraded attitude performance	Star sensors, sun sensor, attitude control electronics	Regularly executed	<ul style="list-style-type: none"> Star sensor signals Sun sensor signals 	<ul style="list-style-type: none"> Sensor outputs recorded over extended period of time Data-fitting routine compares sensor outputs Detectors outside acceptable range are deweighted 	<ul style="list-style-type: none"> Sensor weighting factors

The presence of executive control is evident in the discussions of the selected ACS maintenance functions. These functions often involve deviating from normal subsystem operating procedures to, for example, power unused functional redundant sensors for use in calibrations of on-line equipment. The decision to perform this procedure naturally involves the executive as a party, allowing this temporary change to the normal power budget.

The system executive exercises a level of control appropriate to its function. It monitors, enables and disables ACS activities, but seldom interacts in a substantial way with ACS equipment. This is shown in the discussion of the mode sequencing and momentum management functions. In each case a change in ACS operating mode is involved. But once enabled, these functions execute logic which determines the optimum time to fire a thruster, disable a fault protection algorithm or establish the normal or cruise operating mode. Such real-time control activities are performed best under ACS control.

3.3.3 Autonomous Fault Management Functions

Table III-15 presents a number of functions which give some indication of the logic in a management scheme for the ACS onboard an autonomous earth orbiting three-axis-stable spacecraft. Fault management functions are described which provide protection from a set of generic faults for some standard devices used on earth orbiting spacecraft. Although the individual descriptions are tailored to the device in question, the faults considered and the techniques presented are applicable to a wide range of similar devices. For example, fault protection schemes are given for high, low and noisy signal level abnormalities for a star tracker. These same faults affect any sensor of the sun, earth or other celestial body. The urgency of response to high and low signal abnormalities, as in the case of the star tracker, remains a function of operational mode and use of the sensor for axial control. As another example, protection from the lack of commandability of momentum wheels, leading to high or no response by the wheels, can be applied to a range of actuators including hot or cold gas thrusters and magnetic torquers. In the protection scheme outlined for the momentum wheels, closed loop self tests are performed using the tachometers to check wheel performance. In the case of thrusters or magnetic torquers, attitude sensors may play the role of the tachometers and thus measure the performance of the actuator through structural dynamics and motion effects. Example attitude control algorithms for a generic Celestial Reference Re-acquisition function and Celestial Sensor fault management (Voyager flight algorithm) are included in Appendices B and C respectively.

Table III-15. ACS Autonomous Fault Management Functions (Sheet 1 of 10)

Assembly/Subassembly Functional Description	Generic Fault	Example of Fault/Symptom	System/Subsystem Impact and Criticality	Detection/Sensing	Isolation	Corrective Measures and Effects	Mission Examples
Star tracker/fixed head sensor; bright object shutters; electronics One or two axis attitude reference for three axis stable spacecraft	Signal level abnormality	'High' signal failure	Immediate loss of one or two axis attitude control Eventual loss of pointing control for communication or payload	At multiple of control law execution time, the star magnitude signal is checked against 'bright object' limit Upon TBD failures of check initiate 'flyback & sweep' sequence where sensor is moved within the fixed head After TBD period of time initiate an axial search sequence while maintaining two axis reference, if possible. Also close bright object shutters as appropriate	With failure of 'flyback & sweep' or axial search activation rates dampen initiate re-acquisition of reference If reacquisition fails switch to redundant sensor, electronics and try again to acquire references	Initiation of an axial search or re-acquisition sequence must be accompanied by a payload safing Critical mission needs may force delay in isolation or recovery Switch in sensors will require a recalibration Inertial mode a safe/degraded state	Roll Reference Loss (Viking) Stray Light (Viking Extended Mission) Celestial Sensor/CS+ Logic (Voyager)

*Canopus Star Tracker

Table III-15. ACS Autonomous Fault Management Functions (Sheet 2 of 10)

Assembly/Subassembly Functional Description	Generic Fault	Example of Fault/Symptom	System/Subsystem Impact and Criticality	Detection/Sensing	Isolation	Corrective Measures and Effects	Mission Examples
Star tracker (image dissecting/fixed head sensor; bright object shutters; electronics One or two axis attitude reference for a three axis stable spacecraft	Signal level abnormality	'Low' signal failure	Drift leading to eventual loss of one or two axis attitude control Eventual loss of pointing control for communications or payload	At multiple of control low excursion rate, the star magnitude signal is checked against a 'dark' limit Upon TBD failures of this 'dark' limit check, initiate 'i-lyback and sweep' sequence where sensor is moved within the fixed head After TBD period of time close bright object shutters and initiate axial search sequence while maintaining two axis reference, if possible	With failure of 'flyback & sweep' or axial search activities dampen rates then initiate re-acquisition of references If reacquisition fails switch to redundant sensors or electronics and try again to acquire references	Initiation of an axial search or reacquisition sequence must be accompanied by a payload safing Critical mission needs may force delay in isolation or recovery Switch in sensors will require a recalibration Inertial mode a safe/degraded state	Roll Reference Loss (Viking) Stray Light (Viking Extended Mission) Celestial Sensor/CST Logic (Voyager)

Table III-15. ACS Autonomous Fault Management Functions (Sheet 3 of 10)

Assembly/Subassembly Functional Description	Generic Fault	Example of Fault/Symptom	System/Subsystem Impact and Criticality	Detection/Sensing	Isolation	Corrective Measures and Effects	Mission Examples
Star tracker (image dissecting)/fixed head sensor; bright object shutters; electronics One or two axis attitude reference for a three axis stable spacecraft	Signal level abnormality	Noisy signal	Erratic performance leading to waste of propellant or unwarranted duty level for reaction actuators Loss of pointing control performance	Collect signals (star magnitude) switch high and low gate limits. Average signals; count signals above TAD deviations from the mean Alternately, average axial attitude performance over a TAD interval of time and compare against a stored or past performance level Alternately, compare performance against a redundant sensor(s), as available	With a sufficient number of signals deviating from the mean, switch electronics or sensors	Switching sensors will require recalibration A redundant (functionally) set of sensors may be used if a switch of electronics or trackers fails to correct the problem	

Table III-15. ACS Autonomous Fault Management Functions (Sheet 4 of 10)

Assembly/Subassembly Functional Description	Generic Fault	Example of Fault/Symptom	System/Subsystem Impact and Criticality	Detection/Sensing	Isolation	Corrective Measures and Effects	Mission Examples
Inertial Reference unit/two axis gyros, mounted with an axis in common; electronics Two or three axis attitude control and high rate damping for a three axis stable spacecraft	Signal level abnormality	Redundant channels fail to compare	Failure to dampen high rates may lead to loss of spacecraft Inertial reference a 'safe' state from fault recovery, or during re-acquisition activity Inertial reference unit is used in calibrating optical sensors	Rate or integrated angular signals from two gyros are compared across a common axis, within an established drift tolerance Failure to compare indicates the presence of a fault The gyros must be warm before this comparison test is attempted	A failure leads to a switch to a redundant pair of gyros, as available A further failure of this type with the redundant gyros should lead to a switch in electronics A switch to redundant gyros or electronics should lead to an abort of a maneuver or a restart of an acquisition sequence All but critical payload functions should be aborted and the payload safed with such a failure	This type of test should be disabled during modes where saturation of the gyros is probable, such as launch A switch to redundant gyros or electronics should lead to an abort of a maneuver or a restart of an acquisition sequence All but critical payload functions should be aborted and the payload safed with such a failure	DRIRU fault protection (Voyager)

Table III-15. ACS Autonomous Fault Management Functions (Sheet 5 of 10)

Assembly/Subassembly Functional Description	Generic Fault	Example of Fault/Symptom	System/ Subsystem Impact and Criticality	Detection./Sensing	Isolation	Corrective Measures and Effects	Mission Examples
Inertial Reference Unit/one axis rate damping or two axis rate integrating gyros; redundant electronics Axial rate damping or axial attitude control for a three axis stable spacecraft	Signal level abnormality	Attitude performance anomaly	Possible failure to dampen high rate, leading to loss of spacecraft. Failure in use of inertial reference mode to control pointing accuracy for payload Anomalous re-acquisition of celestial references while in inertial reference mode	Develop an attitude error from a comparison of gyro and optical sensor control algorithms; compare this error against a tolerance for drift, calibration error or operational mode Use timed turns or acquisition sequences, with 'time-out' an indication of an anomaly The gyro must be warm and rate captive occurred before such a performance test is enabled Optical sensors should pass a health check before being used for detection of a gyro fault	A failure leads to a switch to a redundant gyro, as available A switch to redundant electronics is possible given a further failure	A switch to a redundant gyro should lead to an abort of a maneuver or a restart of an acquisition sequence All but critical payload function should be restarted	

Table III-15. ACS Autonomous Fault Management Functions (Sheet 6 of 10)

Assembly/Subassembly Functional Description	Generic Fault	Example of Fault/Symptom	System/ Subsystem Impact and Criticality	Detection/Sensing	Isolation	Corrective Measures and Effects	Mission Examples
Inertial Reference Unit/one axis rate damping or two axis rate integrating gyros; redundant electronics Axial rate damping or axial attitude control for a three axis stable spacecraft	Signal level abnormality	Noisy signal	Erratic performance leading to waste of propellant Failure to perform pointing control at an accuracy needed for payload	Collect rates or angular excursions over a period of time, or during a certain sequence Compute an average and a tolerable deviation from the mean. Use a count of the number of signals above this deviation to determine noise. Compare the collection against those collected during prior excursions of the same sequence Match the collection against stored optimum performance parameters The gyro must be warm and rate capture occurred before the collection of rates or angular excursions begins	With a sufficient number of signals deviating from the mean or from performance parameters switch gyros or electronics	A switch to redundant gyros or electronics which results from failure of this test should be inhibited during critical mission phases. This test and associated redundancy switching should be inhibited during high rate damping associated with acquisition of references. A switch to redundant gyros or electronics should cause an abort of all but critical payload functions	

Table III-15. ACS Autonomous Fault Management Functions (Sheet 7 of 10)

Assembly/Subassembly Functional Description	Generic Fault	Example of Fault/Symptom	System/ Subsystem Impact and Criticality	Detection/Sensing	Isolation	Corrective Measures and Effects	Mission Examples
Potentiometer for an actuator, electronics Tachometer for a reaction wheel; potentiometer for solar array position; potentiometer for an antenna position etc.	Signal abnormality	Incorrect signal	Loss of pointing control due to feedback failure from actuators Loss of power resulting from a failure to drive arrays to proper position Loss of payload functions due to positioning errors	Closed loop self test shows an anomaly: actuator drive signal to potentiometer reading/response Actively redundant potentiometer do not compare to a given tolerance for calibration	Switch to redundant potentiometer or electronics	A redundant potentiometer may need to be recalibrated A switch to a redundant potentiometer may require an abort of a payload sequence	
Solar array drive; electronics Drive array to normal to sun line to power a three axis stable earth orbiting spacecraft	Lack of command-ability	'High' response	Loss of power due to arrays not pointing to the sun line Loss of axial control due to disturbance along axis of the array Loss of sun reference for any array mounted optical sensor, with corresponding loss of axial attitude control	Closed loop self test shows an anomaly: array drive signal to potentiometer reading/response 'Too short' or 'too long' time-out failure in a slew of the array	Switch to redundant drives or electronics	A redundant drive may need calibration Stored momentum resulting from the inadvertent slew may need to be dumped from the system 'Axial' reacquisition may need to be initiated if sensors must recover sun reference Switch to and from battery power may be required during isolation/recovery sequences	

Table III-15. ACS Autonomous Fault Management Functions (Sheet 8 of 10)

Assembly/Subassembly Functional Description	Generic Fault	Example of Fault/Symptom	System/Subsystem Impact and Criticality	Detection/Sensing	Isolation	Corrective Measures and Effects	Mission Examples
Solar array drive; electronics Drive array to be normal to the sun line to power a three axis stable earth orbiting spacecraft	Lack of command-ability	No response	Eventual loss of power due to failure to update array position at orbit rate Degradation/error in axial attitude control due to error introduced in array mounted optical sensors	Closed loop self test shows an anomaly: array drive signal to potentiometer reading/response 'Too long' time-out failure in slew of the array	Switch to redundant drives or electronics	A redundant drive may need calibration A slew can correct array position as part of auto-calibration or time-in-orbit updates from auto-navigation	
Momentum wheels; electronics Storage of axial or bias momentum; reaction attitude control actuator for a three axis stable spacecraft	Lack of command-ability	'High' response	Eventual loss of attitude control Loss of pointing control accuracy for payload Possible loss of celestial references Loss of sensitivity for axial attitude control due to loss of momentum bias	Closed loop self test shows an anomaly: wheel drive signal to tachometer output Optical sensors show anomalous angular excursion or a persistent excursion in one direction Sensors and tachometers must pass a health check before being used for detection of a momentum wheel fault	Switch to redundant drive or electronics, as available Reconfigure among skewed wheels assembly, as possible, stored momentum	A failure to correct by switching to redundant drives/electronics may lead to switching out the failed wheel/drive assembly A momentum unloading may be needed to reduce anomalous stored rates and recover attitude control	

Table III-15. ACS Autonomous Fault Management Functions (Sheet 9 of 10)

Assembly/Subassembly Functional Description	Generic Fault	Example of Fault/Symptom	System/Subsystem Impact and Criticality	Detection/Sensing	Isolation	Corrective Measures and Effects	Mission Examples
Momentum wheels; electronics Storage of bias or axial momentum; attitude control reaction actuator for a three axis stable spacecraft	Lack of command-ability	No response	Eventual loss of attitude control Loss of pointing control accuracy for payload Loss of sensitivity for axial attitude control due to loss of momentum bias	Closed loop self test shows an anomaly: wheel drive signal to tachometer output Optical sensors show angular excursion in one direction Sensors and tachometer must pass a health check before being used for detection of a momentum wheel fault	Switch to redundant drive or electronics, as available Reconfigure among shovels wheels assembly stored momentum, as possible	A failure to correct by switching to redundant drives/electronics may lead to switching out the failed wheel/drive assembly from control	
Processor Execute software logic which represents the control law	Software traps	Memory protect, odd address, illegal instruction	Execution of the control law terminated with loss of spacecraft possible depending on operating mode Loss of attitude control Loss of system data interface to the attitude control subsystem	The processor halts with a software error trap upon the failure of the microcode to execute an instruction Execution of the control law terminated with loss of spacecraft possible depending on operating mode Loss of attitude control Loss of system data interface to the attitude control subsystem	Clear any 'soft' code from memory and reload and restart On next occurrence switch to redundant processor and restart	A software reload requires a reset or reinitialization of the last operating state or mode of the subsystem. In addition some on-going payload activity may need to be aborted A switch in processors must be initiated by a system executive	CCS error (Viking) CCS error (Voyager)

Table III-15. ACS Autonomous Fault Management Functions (Sheet 10 of 10)

Assembly/Subassembly Functional Description	Generic Fault	Example of Fault/Symptom	System/Subsystem Impact and Criticality	Detection/Sensing	Isolation	Corrective Measures and Effects	Mission Examples
Memory Storage for code, write protected data and scratch pad for processing	Memory load failure	Checksum failure on write protected data	Loss of payload or mission related sequencing; possible loss of critical mission objectives	Regularly perform checksum on 'soft' stored data; data words given (any) even parity; data stored in a write protected data	Reload memory from non-volatile storage Switch to redundant memory or processor on recurrence	A memory reload or switch in memory/processor will require a reset or reinitialization of operating mode. In addition some payload activity may need to be aborted and restarted A switch in processor must be initiated by a system executive	
Peripheral interface Electronics interface to subsystem devices or to system data bus from the processor in a subsystem	Interface failure	Data handshake failure, device 'time-out', heartbeat failure	Loss of system data interface to the subsystem Loss of device response in execution of control logic with subsequent loss of attitude control	Periodic transmissions (heartbeats) across system data bus are not received Device fails to indicate receipt of data upon transmission (data handshake) Device fails to respond to request for service within TBD microsecond (time-out)	Switch to redundant interface electronics or redundant devices Repeated failures should lead to switch in processors	A switch in devices may require a re-calibration or other initializing sequence to be scheduled A switch in processors must be initiated by a system executive and a reset or re-initialization of subsystem operating mode required	Heartbeat Generator and Self Test Control (Voyager) AACS Power Code Processing (Voyager) Bad/No Echo Response (Voyager) Catastrophe Handler Processor Faults (Voyager)

3.4* PROPULSION SUBSYSTEM

3.4.1 Functional Description and Elements

Propulsion subsystems for on-orbit applications are primarily monopropellant hydrazine and earth storable bipropellant, typically nitrogen tetroxide-monomethyl hydrazine (N_2O_4 -MMH). Discussion of autonomy as applied to satellite propulsion subsystems will be limited to these two generic chemical types. Both bi-propellant and monopropellant systems have been used on recent planetary missions (Viking and Voyager, respectively) with some autonomy in critical areas on both missions.

3.4.1.1 Propulsion Subsystem Functions. On-orbit propulsion functions fall into two general categories:

(1) Translation

Stationkeeping and orbit adjust maneuvers requiring large delta-V's, usually with steady state firings.

(2) Attitude Control

Acquisition and reacquisition of references, attitude re-orientation, attitude maintenance (e.g. limit cycle pulsing or reaction wheel unloading) and attitude control during translation maneuvers. These functions are usually accomplished in pulse mode (pulse off in the case of some translation maneuvers).

The components and subassemblies of satellite propulsion subsystems are generally not completely redundant because of weight and design considerations; application of autonomy to the propulsion subsystem thus differs in some respects from application to electronic subsystems. Fault management approaches must be tailored to the specific configuration and components. In coping with some faults, degraded operation or alternate operating modes must be selected in lieu of block replacement of failed elements.

3.4.1.2 Subassembly Grouping. Representative subassemblies can be grouped as follows for purposes of discussing options for autonomous operation:

(1) Tanks

(2) Pressurization Components

(a) Regulators

(b) Relief Valves

*By R. W. Rowley

(3) Interconnecting Plumbing

- (a) Lines**
- (b) Filters**
- (c) Check Valves**
- (d) Isolation Valves**

(4) Thrusters

- (a) Translation**
- (b) Attitude Control**

(5) Transducers

Thrusters and isolation valves are usually redundant at least to some degree. Components such as regulators and check valves may be redundant, but components such as tanks, lines and filters are conservatively designed with large safety factors, since redundancy is often impractical.

3.4.2 Autonomous Maintenance Functions

Propulsion subsystem maintenance functions which may have to be made autonomous for extended periods of unattended satellite operation include the following:

(1) Propellant Management

- (a) Mass Used/Remaining**
- (b) Center-of-Mass Location**
- (c) Use Rate/Mission Planning**

(2) Configuration Management

- (a) Isolation Valve Position Monitoring**
- (b) Thruster Selection**
- (c) Tank Selection**

- (3) Navigation
 - (a) Burn Duration Estimates
 - (b) Burn Performance Reconstruction
- (4) Attitude Control
 - (a) Impulse Bit Estimates
 - (b) Impulse Bit Monitoring
- (5) Thruster Life Management
 - (a) Pulse Accumulation
 - (b) Propellant Throughput

Summaries are presented in Table III-16 of the approaches required to perform two representative examples of these functions; 1) Propellant Management, Center-of-Mass Location; and 2) Navigation, Burn Duration Estimates.

Autonomous performance of these functions has not been required to date on any spacecraft; the approaches presented are thus derived from the practices of ground control as they would be incorporated into onboard software.

3.4.3 Autonomous Fault Management Functions

The general categories of fault types encountered in liquid propulsion subsystems for satellite use are summarized in Table III-17. The specific faults to be protected against and the fault management approaches are heavily dependent on the component designs, the propulsion subsystem configuration, and the mission requirements.

A degree of autonomous fault protection has been incorporated in previous planetary spacecraft for the attitude control thrusters (Viking Extended Mission and Voyager) and the regulator (Viking). During the Viking Extended Mission, a routine was also incorporated to terminate the main engine firing at propellant depletion. Autonomous fault protection can be incorporated for other components where the design includes redundancy. For example, isolation valves are usually backed-up such that a failure of any single valve to open or close can be corrected. However, in many applications, low probability failure modes cannot be fully protected against (e.g., tank leakage in single tank systems) and design and test conservatism are used instead of redundant component switching.

3.4.3.1 Fault Sensing Techniques. Techniques for sensing faults are:

- (1) Direct measurement - usually limited to simple pressure, temperature, and valve position sensors in selected locations.

Table III-16. Propulsion Autonomous Maintenance Functions (Sheet 1 of 2)

Generic Function	Example of Maintenance	Scope: Mission/System/ Subsystem Impact	Subsystem Assembly Involved	Frequency of Execution	Inputs	Processing	Outputs
Propellant Management	Determine Mass Remaining and Location of Center-of-Mass	1. Propellant Mass Remaining Required for Navigation: • Maneuver Strategy • Acceleration/Burn Time Estimates	1. Propellant Tanks 2. Pressurant Tanks 3. Movable Elements (Antennae, Instrument Platforms, Solar Panels)	Prior to Orbit Adjust Maneuvers	1. Propellant Tank Temperature and Pressure (Allocated System) 2. Propellant Usage History (Regulated System) 3. Movable Element Position	1. Estimate Mass Remaining in each Tank 2. Estimate Position of Movable Element CH's 3. Estimate Speciecraft Composite CH	1. Propellant Mass Remaining and Total Speciecraft Mass 2. Speciecraft Composite Center-of-Mass Location
		2. Center-of-Mass Knowledge Required for Attitude Control and Navigation • Gimbal Pre-Aim or Duty Cycle Estimates During Maneuver Firings					

Table III-16. Propulsion Autonomous Maintenance Functions (Sheet 2 of 2)

Generic Function	Example of Maintenance	Scope: Mission/System/Subsystem Impact	Subsystem Assembly Involved	Frequency of Execution	Inputs	Processing	Outputs
Orbit Adjust Maneuvers	Estimate Thruster Firing Duration to Provide Required Delta-V	Autonomous Burn Time Estimates Required to Support Autonomous Navigation Capability unless Satellite includes an Accelerometer	1. Propellant Tanks 2. Thrusters	Prior to Orbit Adjust Maneuvers	1. Delta-V Direction/Thruster Selection 2. Delta-V Magnitude Required 3. Thrust Vector Control Thrust Degradation (e.g. Pulse Off Duty Cycle Expected) 4. Propulsion Parameters; Tank Pressure, etc 5. Spacecraft Parameters; Mass, Center-of-Mass	1. Estimate Total Impulse Required To Provide Delta-V 3. Estimate Thrust-Time History including Thrust Vector Control Requirements 3. Select Burn Time	Burn Time Required

Table III-17

LIQUID PROPULSION SUBASSEMBLY FAULT CATEGORIES

<u>Subassembly</u>	<u>Fault Categories (see Note)</u>
Tanks	Leak* - External - Internal (e.g. Diaphragm)
Pressurization Components	Regulator Leakage/ Tank Overpressure
Interconnecting Plumbing	Isolation Valve Fail - Leak or Open - Close External Leak* Filter Clog*
Thrusters	Leak/Fail Open Fail Close/Fail to Operate
Transducers	Faulty Output Signal

Note

Faults marked with an asterisk (*) may not be correctable, depending on propulsion hardware design.

- (2) Inference - reconstruction of subsystem performance by comparing expected and actual parameter changes (e.g., tank pressure decrease, attitude control response, etc.

Limitations in sensor technology currently force a heavy reliance on inferential techniques for propulsion fault detection during both ground based and autonomous operation.

3.4.3.2 Autonomous Fault Management Examples. Summaries of two examples of autonomous fault management techniques are presented in Table II-18.

- (1) Thruster Fault Protection
- (2) Regulator Leakage Protection

Autonomous fault protection algorithms used to date on planetary spacecraft for propulsion related functions are summarized in Part III, Section 2.1.6. These algorithms were:

- (1) Viking - PRSREG - Pressure Regulator Failure
- (2) Viking Extended Mission - CORKER - Automatic Leak Clearing, and ACLMON - Accelerometer Monitor
- (3) Voyager - TCAPUF - Trajectory Correction and Attitude Propulsion Unit Failure

Detailed algorithms for two thruster failure routines, CORKER and TCAPUF, are included in Appendix C.

Table III-18. Propulsion Fault Protection Functions (Sheet 1 of 2)

Assembly/ Subassembly Functional Description	Generic Fault	Example of Fault Symptom	System/ Subsystem Impact and Criticality	Detection/ Sensing	Isolation	Correction	Mission Example
Attitude Control Thruster	Valve Failed Open or Leaking	Sustained Uncommanded Torque	1. Excessive Propellant Consumption 2. Loss of Attitude Control	1. Excessive Pulsing by Opposing Thruster 2. Excessive Wheel Speed Increase 3. Sustained Thrust Chamber Pressure 4. Increasing Thrust Chamber Temperature	Close Upstream Propellant Isolation Valve(s) and Disable Thruster Valve Driver(s)	1. Switch to Backup Thruster 2. Attempt to Clear Leak by Repeated Pulsing	1. TCA V 2. CO V E M
	Valve Failed Closed	Lack of Commanded Torque	Loss of Attitude Control	1. Uncorrected Attitude Drift 2. Lack of Commanded Torque for Wheel Unload or Attitude Maneuver Change Maneuver 3. Zero Output from Thrust Chamber Pressure Transducer 4. Lack of Thrust Chamber Temperature Change	Close Upstream Propellant Isolation Valve(s) and Disable Thruster Valve Driver	Switch to Backup Thruster	TCAP Voy

Table III-18. Propulsion Fault Protection Functions (Sheet 2 of 2)

Assembly/ Subassembly Functional Description	Generic Fault	Example of Fault Symptom	System/ Subsystem Impact and Criticality	Detection/ Sensing	Isolation	Correction	Mission Examples
Pressurant Regulator	Seat Leakage	Overpressure of Propellant Tanks	Relief Valve Venting Reduces Pressurant Gas Supply- Eventual Loss of Propulsion Capability if Pressurant Loss is Excessive (This is a time critical failure if regulator fails wide open)	1. Pressure Increase in Propellant Tank(s) 2. Pressure Decrease in High Pressure Supply Tank(s)	Close Isolation Valve between Pressurant Source and Regulator	1. Operate in Blow- down Mode as Long as Possible 2. Cycle Isolation Valve(s) to Re- Pressurize Propellant Tanks	PRSREG- Viking

3.5.1 Functional Description and Elements

The Thermal Control Subsystem regulates the thermal environment of the spacecraft and its components through a wide variety of active, semi-active, and passive components. Wide use of passive and semi-active components in design applications leads to autonomous operation in a trivial sense that no external control of the components is possible. The primary requirement for additional autonomous control capability arises with active components. Thermal Control Subsystem design is characterized by a high dependence upon the thermal behavior of all other subsystems and their components and the absence of need for welfare maintenance functions.

3.5.1.1 Passive Elements. These are non-moving elements which are not adjustable once installed, although contamination will change their properties.

(1) Thermal Control Surfaces. (Radiators) The thermal optical properties are used to control energy balance.

- (a) Paints
- (b) Metal surface/surface treatment
- (c) Second surface mirrors

(2) Multilayer Insulation (MLI). MLI uses the radiation blocking of many layers of low emissivity surfaces.

- (a) Aluminized Mylar/Kapton
- (b) Metal Foils (Tantalum/Aluminum)

(3) Conduction Straps. Use material with the thermal conductivity required for the application.

- (a) High conductivity - metals (copper/silver/copper)
- (b) Controlled conductivity - (graphite composites)

(4) Thermal Isolation. Use material to minimize thermal conductivity between adjacent equipment.

- (a) Material - Low conductance materials (synthetic foams/composites)
- (b) Design - Contact area control, length

*By R. N. Miyake

3.5.1.2 Semi-Active Elements. The semi-active components are elements that can be ground adjusted, but once in orbit there is no adjustment capability.

- (1) Louvers. Louvers are devices where movable panels change the surface thermal optical properties. The movable elements are driven by bi-metallic elements, and are thus driven directly by temperature.
- (2) Heat Pipes. Heat pipes are vapor chambers which use pressure difference to drive a vapor from the heat source to a sink. These devices usually operate at a very low temperature difference, and are capable of transferring large amounts of energy. Choice of working fluid will define operating temperature level.
- (3) Phase change material (PCM). PCM stores energy at a high generation rate thus reducing temperature rise in a component.

3.5.1.3 Active Elements. These are components which have the capability of thermal control, on-orbit adjustment, and the ability to add energy. Prime examples are:

- (1) Heaters. Convert electrical energy to thermal energy and can be commanded on/off.
- (2) Thermostats. Control (command) electrical power cycling as a function of temperature, and fall into two major categories.
 - (a) Mechanical. A mechanical thermostat is usually a bi-metallic unit that is the switch. The bi-metallic unit can be set to maintain the temperature level and dead band desired.
 - (b) Electronic. This unit uses a sensor output to electronically switch the heater.
- (3) Fluid Loops. Transfer energy between components and sinks by pumping fluid through lines to heat transfer (HT) rejection plates to space sink (radiators).
- (4) Variable Conductance Heat Pipes. Heat pipes with components (heaters, controls, inert gas chamber) that will deactivate the condenser.
- (5) Temperature Controlled Louvers. Similar to the louvers described above, but with heater and electronic thermostat to adjust louver position.

- (6) Active Coolers. Devices that can cool specific areas or components to temperature level required for the successful operation of that area. Two examples are:
 - (a) Dewars. These hold stored low temperature cryogenic fluids that are expended cooling specified equipment.
 - (b) Refrigerators. These are mechanical devices that cool specific areas. This can be done with various devices (adsorption/absorption).
- (7) Temperature Sensors. Measure temperature levels of components.
- (8) Flux Sensors. Calorimeters measure energy incident or absorbed by a surface.

3.5.2 Autonomous Maintenance Functions

No welfare maintenance requirements were identified for the components listed in Topic 3.5.1.

3.5.3 Autonomous Fault Management Functions

The functions identified in Table III-19 cover all components listed in Topic 3.5.1. Several characteristics of thermal control fault management may be deduced from the table:

- (1) Thermal faults are often the result of faulty components in other spacecraft subsystems and may be a symptom of such faults.
- (2) Response to thermal faults often requires reconfiguration of external subsystems.
- (3) This interdependency requires thermal constraints to be considered in the fault management response of a spacecraft executive or another subsystem.

Table III-19. Thermal Control Fault Management Functions (Sheet 1 of 4)

Assembly/ Subassembly Functional Description	Generic Fault	Example of Fault/ Symptom	System/Sub- system Impact & Criticality	Detection/ Sensing	Isolation	Correction	Mission Examples
Passive Components Thermal Con- trol Surface Multilayer Insulation Conduction Strap Thermal Isolation	Contamination	Temperature will go out of predicted range of either high or low	Components out of allowable range (temp.) could cause failure or out of spec operation	Temperature sensing of components Structure temp. sensing Monitor changes in operational characteristics	No isolation capability	Modify operation of S/C comp. to alter energy balance	Viking Mariner & Voyager etc. A.F.SAT.
	Breakage						
	Thermal Leakage						
Semi-active Components Louvers Heat Pipes	Failure of bi-metallic driver Contamination	Change in temp. for a given cond.	Depending on level of fault- from no impact to failure to control temp. of radiator	Temp. sensing of components Structure temp. sensing Temp. sensing of radiator Temp sensing of heat exchange plate, radiator, comp. on heat exch. plate structure	No isolation capability	No on-orbit correction cap. - modify S/C operation	SEASAT A Viking Mariner S/C
	Inert gas contamination	Increase in ΔT between plate and radiator	Depending on extent of cont.- no impact to failure of heat pipe		No isolation capability	No on-orbit correction cap. modify S/C opera- tion	

Table III-19. Thermal Control Fault Management Functions (Sheet 2 of 4)

Assembly/ Subassembly Functional Description	Generic Fault	Example of Fault/ Symptom	System/Sub- system Impact & Criticality	Detection/ Sensing	Isolation	Correction	Mission Examples
Heat Pipes (Cont.)	Loss of Working Fluid	Increase in T between plate and radiator	Leakage will lead to a complete failure of heat pipe	Same as above			
PCM (Phase change material)	Leakage of PCM	Increase in temp. rise	Temp. could go above spec. limits	Temp. sensing of component upon which the PCM is mounted	No isolation capabilities	Change comp. of mode	
Radioisotope Heater	Decay in heat output	Lower temp	Small to no energy output	Temp. sensing of components or RIH	No isolation capability	Change comp. of mode	
Active Components Heaters	Open circuit (failed htr)	Loss of temp. control capability	Usually a single failure will have no effect (Htrs. usually used pri/red on 2 circuits)	Temp. sensing of components Power sensing	Isolate htrs by com- mand (if heaters controlled by command can command off/on or cycle)	Change heaters duty cycle.	Prop line heaters. makeup heaters (Seasat A)
Thermostats Mechanical (Bi-metallic)	Fail Closed (welded points)	Rise in temperature	Over temp. cause out of spec operation	Temp. sensing of component or area (power)	Isolate 1) By off-relay command. 2) Over temp. thermostat failure to turn on (no isola-	Normally O.T. thermo- stat corrects (no commands req'd) Redundant circuit	DSCS III DSCS III
	Fails to turn heater on	Unit cools off	Under temp. cause out of spec operation	Temp sensing of component or area (power)			

TABLE III-19. Thermal Control Fault Management Functions (Sheet 3 of 4)

Assembly/ Subassembly Functional Description	Generic Fault	Example of Fault/ Symptom	System/Sub- system Impact & Criticality	Detection/ Sensing	Isolation	Correction	Mission Examples
Thermostat Electronic	Electronic failure (shift in cal./failure to switch)	Change in temp. level (TIC)	Change in temp. of component	Temp. sensing of component or area (power)	Usual elect. thermostat Voting 3 TH 2 out of 3 if one fail 2 out of 2	3-voting corrects	DSCS III
	Fluid loss	Loss in heat absorption capability	Temp. rise in heat rej. plate	Temp. sensing of component or area (power)	No isolation capability (except re- dundant CKT)	Turn on re- dundant unit	
Fluid Loop	Elect pump failure	Loss in heat rejection cap.	Temp. rise in heat rej. plate. Cooling of radiator	Temp. sensing of component or area/(power) pump	Failure of pump/backup redundant pump on/ redundant CKT	Turn on re- dundant unit	
Variable-Cond. Heat Pipes	Loss of working fluid	Increase in ΔT between plate and radiator	Leakage will lead to a com- plete failure of heat pipe	Temp. sensing of heat exch. plate, radiator, comp. on heat exch. plate, structure	Redundant unit on	Redundant unit	
	Elect.cont. failure (complete) (partial)	Shift in temp. level	If shift is small no effect/ large could fail unit	Same as above plus power for elect. cont.	Redundant unit on.	Redundant unit	
	Inert gas generation	Larger ΔT ?					

TABLE III-19. Thermal Control Fault Management Functions (Sheet 4 of 4)

Assembly/ Subassembly Functional Description	Generic Fault	Example of Fault/ Symptom	System/Sub- system Impact & Criticality	Detection/ Sensing	Isolation	Correction	Mission Examples
Temp. Controlled Louvers	Heater failure	Shift in temp. of component	Could put comp. out of spec temp./op. parameters	Temp. of comp. power cons. of heater	Backup heaters (redundant)	Redundant heaters	
	Contamination	Shift in temp. of component	Could put comp. out of spec temp./op. parameters	Temp. of comp. power cons. of heater	Chg. op.		
Active Coolers Dewars	Heat leak	Depletion of fluid	Loss of comp. operate time	Mass loss detector	No isolation capability	Chg. comp. op. mode	IRAS
Refrigerators	Fluid loss Pump failure Control failure	Rise in temp. of comp.	Detector cannot control temp.	Temp. of detec- tor, syst. power	No isolation capability		
Temp. Sensors	Calshift	Specific temp. unit temp. change	Not critical, sensing only. Critical sensing on control unit	Compare with adjacent T/C	Backup	No on-orbit chg. If shift known, use ground calibration	
Flux Sensors	Calshift	Output change	Not critical - sensing only	Complete with S/C cond. prior to failure & other sensor data	No isolation capability		

Contents

Part IV - Validation Methodology

	<u>Page</u>
1. Validation Philosophies.	IV-2
2. The Validation Process	IV-3
3. Validation Requirements.	IV-3
3.1 Audit Trail Baseline.	IV-3
3.2 Subsystem and System Interfaces	IV-3
3.3 Electromagnetic Interference and Other Transient Phenomena	IV-5
3.4 Diagnostic and Test Software.	IV-5
3.4.1 Diagnostic Software.	IV-5
3.4.2 Test Software.	IV-5
3.5 Mission Unattended Lifetime Demonstration . . .	IV-6
4. Implementation Methods and Techniques.	IV-6
4.1 The Validation/Test Subsystem	IV-6
4.2 Requirements Evaluation and Traceability. . . .	IV-8

PART IV

VALIDATION METHODOLOGY

The validation methodology for autonomous systems assumes an existing baseline program. Autonomy then adds an additional attribute which must be effectively incorporated within existing validation activities.

The following paragraphs list guidelines for modifying the baseline philosophies, processes, requirements, implementation methods and techniques.

SECTION 1

VALIDATION PHILOSOPHIES

Spacecraft subsystems which are required to include autonomous design functions are generally more complex than entire spacecraft systems of the last generation.

This additional complexity, the limited resources available for validation, and the longer expected spacecraft operational lifetimes, dictate that greater emphasis be placed upon early generation and acceptance of design requirements to aid the validation process.

Therefore, it is suggested that the validation requirements on the system design as listed in Part III, Section 2.3 of this document also be reviewed for applicability.

Development of validation philosophies for autonomous systems requires that some existing military and industrial standards and practices be revised to accept an integrated, building block approach for the analysis, test, and simulation of autonomous system functions. In addition, the following paragraphs provide key guidelines for the validation of autonomous spacecraft systems.

Authors: R. Malm and J. Morecroft

SECTION 2

THE VALIDATION PROCESS

The validation process for autonomous system functions begins at the lowest reasonable level of hardware and software assembly. As the elements are assembled toward the full flight configuration, lower levels of autonomous features are no longer available for detailed element revalidation. Therefore, special efforts must be directed toward early engineering demonstration tests of all interfaces and redundant functions.

The system design and the test program must be integrated such that successful test of the higher levels of assembly are an implicit validation of the lower level autonomous functions.

The integrated test plan must clearly demonstrate that autonomous features have been verified several times during the composite test program, and that all critical functional elements are available at launch. Figure IV-1 illustrates such a test flow.

SECTION 3

VALIDATION REQUIREMENTS

The requirements which are significant to the validation of autonomous design are:

3.1 AUDIT TRAIL BASELINE

Elements with autonomous functions shall be tested at the system level with all direct access and monitoring support equipment connected to ensure a completed audit trail of system responses.

These baseline performance data are required for comparison during other test configurations when only telemetry data are available.

3.2 SUBSYSTEM AND SYSTEM INTERFACES

All test sequences shall be directed towards a clear demonstration of how interfacing elements will respond to both nominal and all anticipated abnormal conditions under which the autonomous features are designed to operate.

3.3 ELECTROMAGNETIC INTERFERENCE AND OTHER TRANSIENT PHENOMENA

It is frustrating, risky and expensive to observe unplanned transient events only to discover that inadequate design considerations, environmental conditions, or test configurations have triggered false failure detection and correction actions. To reduce the risk of test failure the following requirements apply:

- (1) Accurate modeling of EMI and environmental transients shall be provided for all normal and abnormal conditions.
- (2) Good cabling and grounding practices shall be observed in all detail designs and test configurations to avoid false actions of autonomous devices.

3.4 DIAGNOSTIC AND TEST SOFTWARE

Software system designs shall provide diagnostic and test software to rapidly isolate failures to the level of the replaceable functional element and to rapidly establish and reconfigure a test sequence. This requires a change to the generally accepted practice that only flight software is to be used for the test program. Whether the software resides in the flight or ground segments is a matter of economic and engineering judgement.

Carefully designed diagnostic and test software should be used to test autonomous features rather than the higher risk method of inducing faults through insertion of interface breakout boxes or cabling.

The very act of creating such software provides an insight into the real software/hardware interfaces which is otherwise achieved only with several software revisions as the result of test failures.

3.4.1 Diagnostic Software

Diagnostic software shall be provided to stress timing constraints, loading, memory limitations and data transfer protocols for all subsystem and system level interfaces.

3.4.2 Test Software

Test software shall be provided to rapidly sequence through mission configurations, mission events, system/subsystem logical states and all data exchange formats.

Test software shall include the ability to corrupt data, messages, frames, fields, words, and bits to validate proper autonomous algorithm actions.

Test software shall provide the ability to temporarily inhibit autonomous actions while at the same time executing the autonomous algorithm up to the point of command execution outputs.

In all cases where diagnostic or test software is provided, an automatic reset to the nominal configuration shall be provided to avoid the inadvertent defeat of autonomous features.

3.5 MISSION UNATTENDED LIFETIME DEMONSTRATION

A test sequences of a duration equal to the minimum period of unattended operations is required. These sequences may be run during spacecraft acceptance tests or flight operations test and training activities.

- (1) During execution of these sequences a minimum of ground support equipment shall be connected to the spacecraft.
- (2) All autonomous functions shall be exercised in such a way that the available functional redundant elements share the operating time equally. (+100 hours is acceptable.)
- (3) All redundant element switching logic shall be demonstrated, starting from a baseline of both prime and redundant element equipment.

SECTION 4

IMPLEMENTATION METHODS AND TECHNIQUES

4.1 THE VALIDATION/TEST SUBSYSTEM

The validation requirements for conventional spacecraft system designs are generally well documented and will not be discussed here.

Autonomous Systems Validation, however, requires new tools and methods, if schedules, risks, and cost are to remain equivalent to those of conventional spacecraft designs.

One of the methods used to include validation considerations in the design process is to identify and establish a validation/test subsystem as part of the flight and ground segment functional designs.

The Validation/Test subsystem functions identified could then be:

- (1) Distributed among existing subsystems on an available resource basis,

- (2) Provided in a common processor for both ground and in-flight use,
- (3) Assigned to one of the flight spare subsystem elements, or
- (4) Assigned to the ground system through a flight system interface.

As a minimum the flight system validation/test subsystem could provide a direct access path to the flight system for ground support equipment interfaces.

The ground support equipment could then provide the necessary tools and processing for system level validation/test.

The top level functional requirements for a validation/test subsystem are to:

- (1) Provide for external interfaces to ground support equipment remote terminals.
- (2) Provide reference for timing flight and ground system element events.
- (3) Provide for corruption of interface protocol and data.
- (4) Provide for simulation/emulation of subsystem interfaces when subsystems are removed for problem investigation or repair.
- (5) Provide for measurement of subsystem processing performance margins.
- (6) Provide for measurement of subsystem timing accuracies.
- (7) Provide interface diagnostics and a test data generator for each subsystem.
- (8) Accept, process and summarize status, alarm event and error information from all subsystems.
- (9) Provide for the periodic transmission of collected information to the flight telemetry subsystem and/or the ground support equipment.
- (10) Provide for monitoring of subsystem responses to command message requests, and provide reports of responses outside established limits (watchdog timer).
- (11) Provide for periodic self-test of the validation/test subsystem elements and non-active spare subsystems.

- (12) Provide for a 50% margin in validation/test resources to accommodate changes after flight subsystem designs are complete.
- (13) Provide test messages for system/subsystem protective coding evaluation to aid in the isolation of faults to the level of the replaceable spares.
- (14) Provide for the development of all software validation/test functions through an independent agency implementation using only project-controlled interface and design documents. This independent implementation will help identify documentation errors and ambiguities.

Early project acceptance and funding of a validation/test subsystem are essential in the era of autonomous system designs.

New technologies (i.e., fault tolerant integrated systems) may reduce the need for a validation/test subsystem hardware/software. However, the explicit collection of test requirements is essential for autonomous systems where faults may be overlooked because of automated fault correction techniques.

4.2 REQUIREMENTS EVALUATION AND TRACEABILITY

One of the most useful techniques for understanding the impact of autonomous functional and design requirements on the validation program is the Requirement Traceability Table.

As an example, Table IV-1 lists:

- (1) The requirements for Part III, Section 2.3, and for Part IV, Section 3 of this document,
- (2) The recommended validation methods (analysis, test, or simulation),
- (3) The assembly levels (module, subsystem, system) at which the requirement can be validated,
- (4) The relative priorities for allocation of validation resources,
- (5) The source of the requirement for additional information or rationale, and
- (6) Comments relating to schedules, test frequency and test tools required.

Table IV-1. Validation Requirements Traceability (Sheet 1 of 4)

REQUIREMENT	VALIDATION METHODS (1)	VALIDATION LEVELS	RELATIVE PRIORITY FOR VALID.	SOURCE REFERENCE	COMMENTS
Inhibit Autonomous Functions	T	s/s,s	3	III: 2.3.1, (9) [Goals: B,(1),; C,(2)]	Required for rapid initialization & test recovery
Command Functions Self-Test/Checksum	T	s/s,s	2	III: 2.3.1, (10) [Goals: A,(3); C,(1)]	
Prohibit Toggle Commands	A,T	s/s,s	1	III: 2.3.1, (11) and NASA command standard [Goals: B,(1),(3)]	
Checkpoint Critical State	T	s/s,s	3	III: 2.3.1, (12) [Goals: A,(2)]	
Event and Status Data	T	s/s,s	3	III: 2.3.1, (13),(14) (15) [Goals: A,(4),(5)]	
Validation Command Philosophies	T	s/s,s	2	III: 2.3.1 (15) [Goals: A,(5)]	
Interface Documentation	T	s/s,s	2	III:2.3.1 (17)	

LEGENDS: Validation Methods: A=Analysis, T=Test, S=Simulation

Validation Levels: m=module, s/s=subsystem, s=system

Priorities: 1) Essential for validation/mission operations. 2) Highly desirable. Results in substantial reduction of operational complexity/staffing/risk. 3) Desirable. Results in reduction of operational complexity/staffing/risk, however, work-arounds are possible.

Source Reference: III=Design & Validation Methodology Guidelines, Part III

Table IV-1. Validation Requirements Traceability (Sheet 2 of 4)

REQUIREMENT	VALIDATION METHODS (1)	VALIDATION LEVELS	RELATIVE PRIORITY FOR VALID.	SOURCE REFERENCE	COMMENTS
Test Hardware Interfaces	N/A	s/s,s	1	III: 2.3.1.(18) [Goals: B,(1)]	QA inspection and certification required
Audit Trail Time	T	s/s,s	1	III: 2.3.2.1(1) [Goals: A,(4),(5)]	
Audit Trail Latest Available Data Table	T	s	2	III: 2.3.2.1.(2),(3),(4),(5),(6),(7) [Goals: A,(4),(5)]	
Audit Trail Storage Requirements	A	s/s,s	2	III: 2.3.2.2.,(1),(2),(3),(4),(5) [Goals: A,(4),(5)]	
Audit Trail Baseline	T	s	2	IV: 3.1	Might be performed at s/s level
Subsystem & System Interface Perf.	T	s/s,s	1	IV: 3.2 [Goals: A,(3)]	

LEGENDS: Validation Methods: A=Analysis, T=Test, S=Simulation

Validation Levels: m=module, s/s=subsystem, s=system

Priorities: 1) Essential for validation/mission operations. 2) Highly desirable. Results in substantial reduction of operational complexity/staffing/risk. 3) Desirable. Results in reduction of operational complexity/staffing/risk, however, work-arounds are possible.

Source Reference:

III=Design & Validation Methodology Guidelines, Part III

IV=Design & Validation Methodology Guidelines, Part IV

Goals=Goals for Air Force Autonomous Spacecraft, SD-TH-81-72, Part VI

Table IV-1. Validation Requirements Traceability (Sheet 3 of 4)

REQUIREMENT	VALIDATION METHODS (1)	VALIDATION LEVELS	RELATIVE PRIORITY FOR VALID.	SOURCE REFERENCE	COMMENTS
Autonomous Control Initiation	T	s/s,s	3	III: 2.3.2, (1) [Goals: B,(1)]	Breakout box initiation shall be avoided
Autonomous Status Indicator	T	s/s,s	3	III: 2.3.1, (2) [Goals: A,(4)(5)]	
Periodic Self Test	A,T,S	m,s/s,s	3	III: 2.3.1, (3) [Goals: A,(5); B,(3)]	
Safe-Hold Only With Catastrophic Event	A,S	s/s,s	1	III: 2.3.1,(4) [Goals: A,(4),(5)]	Test may create high risk (PTM only)
Fault Recovery Criteria	T	s/s,s	2	III: 2.3.1,(5),(6) [Goals: B,(1),(2)]	
Fault Management Processing	T	s/s,s	3	III: 2.3.1, (7) [Goals: B,(3)]	
Time Tag Resolution	T	s/s,s	1	III: 2.3.1, (8) [Goals: A,(4)(5)]	

LEGENDS: Validation Methods: A=Analysis, T=Test, S=Simulation

Validation Levels: m=module, s/s=subsystem, s=system

Priorities: 1) Essential for validation/mission operations. 2) Highly desirable. Results in substantial reduction of operational complexity/staffing/risk. 3) Desirable. Results in reduction of operational complexity/staffing/risk, however, work-arounds are possible.

Source Reference: III=Design & Validation Methodology Guidelines, Part III

IV=Design & Validation Methodology Guidelines, Part IV

Table IV-1. Validation Requirements Traceability (Sheet 4 of 4)

REQUIREMENT	VALIDATION METHODS (1)	VALIDATION LEVELS	RELATIVE PRIORITY FOR VALID.	SOURCE REFERENCE	COMMENTS
EMI & Transient	A,T,S	s/s,s	1	IV: 3.3 [Goals: A,(3)]	Time compression is the minimum requirement May be divided into several consecutive test sequences
Diagnostic & Test Software	T	s/s,s	3	IV: 3.4 (all) [Goals: A,(4);B,(1)]	
Lifetime Demonstration	T	s	2	IV: 3. [Goals: A,(1)]	

LEGENDS: Validation Methods: A=Analysis, T=Test, S=Simulation

Validation Levels: m=module, s/s=Subsystem, s=system

Priorities: 1) Essential for validation/mission complexity/staffing/risk. 2) Highly desirable. Results in substantial reduction of operational complexity/staffing/risk, however, work-arounds are possible. 3) Desirable. Results in reduction of operational complexity/staffing/risk, however, work-arounds are possible.

Source Reference: III-Design & Validation Methodology Guidelines, Part III

IV-Design & Validation Methodology Guidelines, Part IV

Goals=Goals for Air Force Autonomous Spacecraft, SD-TR-81-72, Part VI

Contents

Part V - Summary of Experience with Autonomous Design and Validation

	<u>Page</u>
1. Background	V-3
2. Project Level Details.	V-4
2.1 Voyager Project Policies	V-4
2.2 Galileo Project Policies	V-4
3. Design Implementation Experience	V-6
4. Validation Experience with Autonomy	V-8
4.1 Background	V-8
4.2 Key Philosophies	V-10
4.2.1 Validation Organization.	V-10
4.2.1.1 Subsystem Testing and Test Sequences	V-10
4.2.1.2 Personnel Responsibilities	V-10
4.2.1.3 Delegation of Responsibility	V-11
4.2.1.4 Hardware/Software Removal	
Authorization.	V-11
4.2.1.5 Hardware Handling Restrictions	V-11
4.2.1.6 Quality Assurance Responsibilities	V-11
4.3 Mission Validation Objectives and Priorities	V-11
4.4 Qualification and Flight Acceptance Testing.	V-11
4.4.1 Test Levels.	V-11
4.4.2 Performance Levels	V-12
4.4.3 Retest Requirements.	V-12
4.5 Levels of Testing.	V-12
4.5.1 Lowest Reasonable Level.	V-12
4.5.2 Test Repetition.	V-12
4.6 Engineering Tests.	V-12
4.6.1 Importance of Early Testing.	V-12
4.7 Flight Equipment Operating Time.	V-13
4.7.1 Minimum and Maximum Operating Time	V-13
4.7.2 Test Value Selection	V-13

	<u>Page</u>
4.8 Regression Testing	V-13
4.8.1 Test Requirements for Replacement Hardware . . .	V-13
4.8.2 Changes, Repairs and Replacements.	V-13
4.8.3 Interface Revalidation	V-13
4.9 System Tests	V-13
4.9.1 Worst-Case Sequences	V-13
4.9.2 Anticipated Environments	V-14
4.9.3 Environmental Testing.	V-14
4.9.4 Audit Trail Verification	V-14
4.9.5 System Test Data Analysis.	V-14
4.10 Subsystem Tests.	V-14
4.10.1 Simulator Requirements.	V-14
4.10.2 Interface Testing	V-15
4.11 Post-Environmental Inspection.	V-15
4.12 Test Plans and Related Documents	V-15
4.13 Test Facilities.	V-16
5. Autonomy Implementation Recommendations.	V-17

PART V*

SUMMARY OF EXPERIENCE WITH AUTONOMOUS DESIGN AND VALIDATION

SECTION 1

BACKGROUND

This Part of the Handbook provides a concise summary of "lessons learned" from previous JPL experience with autonomous spacecraft operating features. The material is culled from presentations, papers, and discussions with key personnel from the Viking, Voyager, and Galileo projects. Galileo related material reflects design policies and approaches that were adopted as a result of previous Viking and Voyager experience and a general JPL design methodology.

*By R. W. Rowley and P. R. Turner

SECTION 2

PROJECT LEVEL DETAILS

Autonomy considerations have been driven by a series of Project level policies.

2.1 VOYAGER PROJECT POLICIES

The overriding requirement on the Voyager autonomy capability was to support the project reliability requirement to eliminate any single point failure which could cause loss of more than 50% of engineering data or the data from more than one science payload instrument. Design considerations for on-board operations were dominated by the long communications delays (up to 90 minutes one way light time at Saturn) and separation of tracking support periods by as much as 24 hours during cruise phases of the mission.

Priority was given to critical functions resulting in the bulk of the Voyager autonomy being applied to the following areas:

- (1) Ensuring spacecraft safety.
- (2) Maintaining spacecraft power.
- (3) Maintaining uplink (command) and downlink (data) capabilities. This required maintaining attitude reference for antenna pointing as well as insuring a functional complement of telecommunication equipment.
- (4) Minimizing consumables (hydrazine) usage.

The Voyager spacecraft mission and functional design are described in Appendix A.

2.2 GALILEO PROJECT POLICIES

The Galileo mission to place an orbiter about Jupiter and deliver an atmospheric probe has continued the basic single point failure policy of Voyager. In addition, policies have been developed for orbiter fault protection and computer margin management.

The fault protection policy calls for an explicit "fault protection subsystem" and defines the modes of operation in each mission phase. The fault protection subsystem is not defined as a separate hardware subsystem, but will be implemented through software control of existing command, data processing, and attitude control resources. The policy recognizes fault protection as an explicit function to be provided in design and given high level management visibility.

The computer margin management policy identifies required memory and timing margins at different milestones from preliminary requirements

review through flight operations. Provision of capabilities over and above minimal requirements is necessary to prevent software design and software changes during operations from being constrained by lack of capacity. The importance of software in the spacecraft designs, the length of the mission (approximately eight years from requirements review to end of mission), and a policy to develop additional software in flight all contribute to the need to provide performance margins in the design.

SECTION 3

DESIGN IMPLEMENTATION EXPERIENCE

Autonomous control and fault management experience has provided a point of departure for Project Galileo and other design activities. Key observations of spacecraft designers have been:

- (1) Voyager autonomous features that have executed in flight have always performed as designed. Frequently, however, fault protection routines were required to function under conditions not anticipated during design and test. The result would be an unexpected series of events which would not threaten the spacecraft, but puzzle ground controllers. Reconstruction of events was complicated by lack of a comprehensive audit trail. This indicates the importance of considering all aspects of validation and flight operations in the design phase.
- (2) Fault protection algorithms generally did not require an independent confirmation of faults before being triggered. The spacecraft was thus not well protected against sensor failures.
- (3) In-flight software evolved as experience was gained, both before and after launch. The advantages of a flexible software design (table driven; small, modular routines) were apparent throughout the development, test, and flight operations phases as frequent changes were made.
- (4) Availability of a baseline hardware design with Failure Modes and Effects Analysis (FMEA's) at the start of the fault protection software design stage resulted in a logical progression from design through software development with minimal iteration. However, the software design was severely constrained by being forced to fit in the small amount of memory available at that stage of the spacecraft design.
- (5) Both formal and informal reviews of the software designs proved to be invaluable. Informal reviews took the form of peer review (individual and group) "brain picking" and "what if" sessions that often disclosed fatal flaws not initially obvious.
- (6) Fault routines produced extensive switching of cross-strapped and block redundant elements. While no problems have been apparent to date, concern remains that switching should be more constrained in future designs.

- (7) Fault routines were triggered often during early stages of the mission. This was frequently caused by thresholds and tolerances being set too tight in the software. Experience indicated that software trip points be set as loosely as is tolerable from an operational standpoint until the spacecraft is checked out in flight. Trip points can then be progressively tightened as operating experience accumulates.
- (8) Input data for triggering fault routines should be filtered by multiple sampling or some other technique to prevent transient conditions from improperly activating the routine.

SECTION 4*

VALIDATION EXPERIENCE WITH AUTONOMY

4.1 BACKGROUND

The philosophies which guide the planetary spacecraft validation process originated during the development of the Corporal and Sergeant missile systems for the U. S. Army, during the mid-1950's. As the lunar and planetary missions of the 1960's and 1970's were developed, the spacecraft became more complex with complicated interfaces, sophisticated systems and subsystems, and the added features of functional redundancy. Each successive spacecraft design has influenced and extended the earlier validation philosophies and methodology. The Viking Project initiated the on-board software autonomous capabilities with fault detection and fault correction routines. These initial autonomous capabilities were then augmented for the Viking extended mission to include on-board routine maintenance. The Voyager Project added additional autonomous features which again extended the scope and complexity of the validation process.

Because autonomy is generally implemented in an on-board computer, via software, validation methodology had to be expanded to include simulation of fault conditions and monitoring of related software operation and actions. Proper interaction between normal and abnormal spacecraft activities was another characteristic of these autonomous systems which had to be validated.

Table V-1 lists the on-board software routines for Voyager and Viking spacecraft, with appropriate supplemental test information for each. This table, and the related following comments, show the manner in which validation philosophies and methodology had adapted to the advent of autonomous spacecraft.

For those routines available during system test, three test environments were used:

- (1) Some of the routines were considered to be normal constituents of spacecraft operation and their validation was a by-product of normal test procedures. In fact, the Voyager ERROR routine was used as a convenient means of stopping test sequences when necessary.
- (2) Most of the others were tested individually, by configuring the spacecraft appropriately and activating the trigger for the routine under test. Results were observed with ground support equipment connected to monitor proper system operation.
- (3) The remainder were validated during execution of spacecraft functional command sequence blocks to which the routines relate. Additional tests were run with some of the routines while executing critical mission sequences. These tests were designed to establish that sequences would proceed properly after autonomous fault correction actions.

*By R. Malm and J. Morecroft

Table V-1. System Level Test History

On Board Routines	Voyager	Viking	Viking Extended	Tested in Test System	Spacecraft Mode(s) for Test	Not Tested in Sys Test	Developed After Launch
Command Loss (CMDLOS)	X			X	CRUISE		
RF Loss (RFLOSS)	X			X	SPECIAL		
Power Check (PWRCHK)	X			X	TCM, ENC		
IRIS Power (IRSPWR)	X			X	ENC		
AACS Power Codes (AACSIN)	X			X	ALL		
Error	X			X	TCM, ENC		
Tandem & Turn Support (TRNSUP)	X			X	TCM		
AACS FCP SWAP (Heartbeat)	X			X	SPECIAL		
Omen Power Code Response	X			X	SPECIAL		
Celestial Loss/Acquisition	X			X	LNCH, CRU		
Power Supply Fail	X					X	
Memory Refresh Fail	X					X	
Thruster Branch Fail	X			X	TCM		
Gyro Fail	X			X	TCM		
Scan Slew Fail	X			X	ENC		
Command Parity Fail	X					X	
Command Sequence Fail	X					X	
Bad/No Echo Response	X			X	SPECIAL		
TCM Turn Abort	X			X	TCM		
Turn Complete	X			X			
Telemetry	X			X			
Backup Mission Load	X					X	X
Telemetry		X		X			
CCS Error		X		X	SEVERAL		
MOI Power transient (MOIMAU)		X		X	MOI		
RF Power Loss		X		X	SPECIAL		
Command Loss		X		X	CRUISE		
Roll Reference Loss		X		X	CRUISE		
Sun Acquisition		X		X	LAUNCH		
ACE Power Changeover		X		X	SPECIAL		
Battery Over-Temp		X		X	CRUISE		
Share Mode		X		X	CRUISE		
Pressure Regulator Leak		X				X	X
Telemetry			X			X	X
(BAT MON)			X			X	X
(BATCHARGE)			X			X	X
(SIN PON)			X			X	X
Receiver Switch (RCVRSW)			X			X	X
Downlink Off (DLOFF)			X			X	X
Seal Leaky Valve (CORKER)			X			X	X
Accelerometer Monitor (ACLMON)			X			X	X
Leak Check (LEAKCK)			X			X	X
Stray Light Response (STRAY)			X			X	X
(DECOM)			X			X	X

ENC - ENCOUNTER

LNCH - LAUNCH

MOI - MARS ORBIT INSERTION

TCM - TRAJECTORY CORRECTION MANEUVER

As noted in the table, some of the available routines were not validated during system test. These are not testable because of their nature, constraints in the system test environment, or because the routine was developed in the post-launch period.

The philosophies and methodology related to validation of autonomous spacecraft, which are the outgrowth of Viking and Voyager experience, contributed significantly to the success of these missions and are equally applicable to other missions employing autonomous spacecraft. They will be the basis for validating the autonomous capabilities of the Galileo spacecraft.

4.2 KEY PHILOSOPHIES

A summary of the planetary spacecraft validation philosophies is given in the following paragraphs:

It should be noted that many of the design characteristics listed in other parts of this document are an essential part of the validation program, therefore, the reader should become familiar with those elements of the system design that are essential for implementation of an effective validation program.

Two (2) general philosophies applicable to all levels of the planetary spacecraft validation program are:

- (1) Validation by testing is preferable to validation by analysis or simulation. Analysis and/or simulation are used only where testing is not practical, or is precluded on the basis of cost, risk, schedule, resources, and facilities.
- (2) Lower level tests need not be repeated once an element has been completely integrated into a higher level configuration unless trouble shooting or periodic trend analysis is not otherwise possible.

Design of higher level tests must be such that lower level function and requirement conformity is implicit in satisfactory completion of higher level tests.

4.2.1 Validation Organization

4.2.1.1 Subsystem Testing and Test Sequences. Subsystem testing, prior to subsystem integration, is the responsibility of the source organization as negotiated with the project. Written subsystem test sequences form the basis for system level tests.

4.2.1.2 Personnel Responsibilities. Subsystem integration, flight system testing and flight system operations are planned and executed by project teams consisting of key personnel from all project, mission, system, and subsystem

4.2.1.3 Delegation of Responsibility. System test responsibility and the required authority to plan, schedule, execute, direct and control the system test program are delegated by the project to the test manager/test conductor.

4.2.1.4 Hardware/Software Removal Authorization. Authorization for removal and replacement of system hardware and software is at the discretion of the test manager/test conductor.

4.2.1.5 Hardware Handling Restrictions. Hardware handling, for system level testing, is specifically restricted to designated team mechanical, electrical, and quality assurance personnel.

4.2.1.6 Quality Assurance Responsibilities. Quality assurance and reliability personnel monitor all flight system test activities, and control flight hardware handling and storage.

4.3 MISSION VALIDATION OBJECTIVES AND PRIORITIES

Objectives and priorities of the mission validation process, as implemented in typical deep space missions, are:

<u>Priority</u>	<u>Objective</u>
1	To demonstrate that each flight system element and the support systems satisfy their mission-related, functional, performance, and operational requirements, while operating in the simulated/actual mission environment(s).
2	To demonstrate that each flight system element, ground systems, and support systems satisfy their interface requirements to all Interface Control Documents (ICDs) and other pertinent documents.
3	To demonstrate that all project elements, i.e., flight system, ground systems, support systems, operational personnel, support facilities, etc., operating together through timed sequences covering all segments of the expected range of mission profiles, can accomplish all mission objectives. That is, that all project elements working in unison can satisfy the mission-level, functional performance, and operational requirements.

4.4 QUALIFICATION AND FLIGHT ACCEPTANCE TESTING

4.4.1 Test Levels

Two levels of test severity are defined as Qualification Tests; 1) Type Approval Test and 2) Flight Acceptance Tests. In general, the test conditions for Qualification Test are fifty percent more rigorous than the nominal anticipated flight conditions. The latter are generally the basis

conducting Flight Acceptance Tests. Acceptable performance limits may be greater under Qualification Test conditions.

4.4.2 Performance Levels

Both types of tests are performed, as appropriate, at hardware piece part and assembly levels. Subsystem level testing is the most extensive, with functional, handling, storage, and operating environmental tests performed. A subset of these tests is performed after integration, as part of the system test procedures, to verify performance of critical elements in the environmental conditions resulting from their location in the flight systems.

4.4.3 Recast Requirements

Subsystems may have to be retested, both Qualification and Flight Acceptance, to new environmental requirements in the event that system tests determine that the operating environment in the system configuration is more severe than that under which that subsystem was originally tested.

4.5 LEVELS OF TESTING

4.5.1 Lowest Reasonable Level

Tests are conducted at the lowest reasonable level of hardware and software assembly. (Component, subassembly, assembly, subsystem, system, and mission levels).

4.5.2 Test Repetition

Functional tests, interface, performance, and calibration sequences are periodically repeated at each assembly level as the equipment is assembled to the flight system launch configuration.

4.6 ENGINEERING TESTS

4.6.1 Importance of Early Testing

All complex, critical hardware, software functions and interfaces are tested as early in the test program as possible using flight equipment, flight prototype or engineering model deliverables.

4.7 FLIGHT EQUIPMENT OPERATING TIME

4.7.1 Minimum and Maximum Operating Times

Minimum and maximum equipment operating times are established at all test levels.

4.7.2 Test Value Selection

Maximum values are selected to detect infant mortality failures. Maximum values are selected to allow completion of system level testing with the set of limited life equipment to be launched.

4.8 REGRESSION TESTING

4.8.1 Test Requirements for Replacement Hardware

All replacement system hardware is subject to the same inspection, handling and acceptance tests as are required during initial installation.

4.8.2 Changes, Repairs and Replacements.

Software version changes or hardware repairs or replacements require a detailed regression test of all affected functions and interfaces.

4.8.3 Interface Revalidation.

Interfaces are not revalidated unless physically disturbed (connectors demated) or redesigned. Redesigned interfaces require complete revalidation. Revalidation of demated connectors is limited to connector integrity and functional test of affected circuits.

4.9 SYSTEM TESTS

4.9.1 Worst-Case Sequences.

Selected worst-case spacecraft engineering and payload mission sequences are provided for system level test to verify that spacecraft performance will meet mission sequence requirements.

4.9.2 Anticipated Environments.

Anticipated flight system environments are included as part of all appropriate system level tests (RFI, static discharge, shock, etc.). System level environmental tests complement but do not replace subsystem environmental test sequences. Hazardous environmental testing is limited to the subsystem level (i.e., thruster firings). Subsystems with hazardous elements provide functional and electrical simulators for use during system level tests.

4.9.3 Environmental Testing.

System level thermal vacuum and vibration tests are conducted as early in the test program as possible. Consistent system performance under ambient conditions must be demonstrated prior to start of environmental test sequences. Failures or hardware replacements following system environmental tests may require that test segments be repeated at the system or subsystem level.

4.9.4 Audit Trail Verification.

Elements with autonomous functions are tested in system test configurations with all direct access and monitoring support equipment to ensure a complete audit trail of system responses.

4.9.5 System Test Data Analysis

Real time displays provided for system test nominally display data in suppressed message formats. Data alarm limits are also provided to alert operations personnel to data excursions beyond planned values.

Fixed periodic status summary formats, the real time data and measurement plots are provided for non-real time trend analysis.

4.10 SUBSYSTEM TESTS

4.10.1 Simulator Requirements.

Functional and electrical simulators are required for all subsystems with complex external interfaces or for subsystems interfaces which are required for other elements to functionally operate.

These simulators must be representative of subsystem designs for interface and circuits, critical timing circuits and ground circuits.

4.10.2 Interface Testing.

Internal and external interfaces are tested as early as the implemented interfaces are available. These tests are repeated several times as the flight testing proceeds toward launch.

4.11 POST-ENVIRONMENTAL INSPECTION

Easy removal and replacement of hardware elements has long been a characteristic design feature of planetary mission systems and subsystems. The Post-Environmental Inspection Period represents the most extensive disassembly of a validated flight system permitted in the test program. In this period, hardware elements and subsystems are available for quality assurance system and subsystem inspection, and calibration and trend analysis which cannot be completed in the flight system configuration. Particular attention is directed toward assessment of environmental and handling damage which may have occurred since initial delivery. The value of this approach has been demonstrated on numerous occasions when contamination, calibration drifts, and subsystem intermittents were detected and repaired. System re-assembly is followed by a repeat of the complete system test, with particular emphasis on equipments which were repaired and replaced. In some instances, equally qualified spare units are used to replace items which cannot be reworked without significant schedule impacts.

It is understandable that such a disassembly would increase potential risks for systems which were not designed for removal and replacement. However, autonomous planetary systems are designed from the start to allow interchangeability of spares without serious concern about costly and detailed re-validation beyond the system interfaces.

4.12 TEST PLANS AND RELATED DOCUMENTS

Implicit or explicit test plans are formulated at all validation levels. These plans result in formal and informal deliverable documentation at each delivery point. This documentation extends from specific test plans to peripheral documents defining necessary hardware and software characteristics derived from implementing the test plan. Supplemental deliverable documentation includes:

- Interface circuit data sheets
- Grounding tree diagrams
- Subsystem/System test procedures
- Problem/Failure reports
- System/Subsystem data flow diagrams
- State diagrams

Each of these documents provides significant contributions to a successful validation program.

The value of validation documentation will be enhanced by adhering to the following principles in its use:

- (a) Written procedures must be generated for all test sequences.
- (b) Delivered documentation must be validated prior to use, just as is the case for hardware and software.
- (c) Appropriate written requests for special tests serve as documentation for such tests.
- (d) Formal failure reports are required for all identified documentation deficiencies, as well as for those of hardware and software.
- (e) Cognizant engineering management must review and consent to all failure report closures.

4.13 TEST FACILITIES

The environment in which system level assembly and test are conducted must be the cleanest available, consistent with mission requirements.

Cleanliness shall be sufficient such that autonomous functions need not be designed for continuous protection against contamination products generated in pre-launch testing or in post-launch operations.

SECTION 5

AUTONOMY IMPLEMENTATION RECOMMENDATIONS

The following points have been selected from a review of previous experience. These represent details which were included in successful designs or would have enhanced the designs if they had been applied.

- (1) Fault management is an overall spacecraft system function. It should be a specific functional design responsibility to be integrated with system and subsystem design.
- (2) In general, block redundancy is a more economical means of supporting reliability goals at the subsystem level than functional redundancy.
- (3) Design for autonomous operations must allow for traceability of autonomous actions after the fact. Logical paths entered, triggering conditions, memory contents and associated hardware status are needed.
- (4) Validation requirements on spacecraft design should be included in the spacecraft design requirements as early as possible. These requirements should include detailed design practices and operating requirements that have been proven necessary for proper validation of spacecraft system and subsystem operation.
- (5) Validation and flight operations experience produced new spacecraft requirements that could not be perceived in the design process. It is important to provide the spacecraft with capability margins above those anticipated in design requirements. Memory margin for spacecraft computers is particularly important, as software is the cheapest implementation of requirement changes after integration and the only implementation after launch.
- (6) The basis of any autonomous control feature is the ability of a control authority to sense a state or condition and command a change. Provision for a processor to access engineering telemetry and initiate command actions allows for the design of software to control a wide variety of spacecraft functions that might not initially require autonomous control.
- (7) Table driven logic software designs have shown 10 to 1 improvements in development cost over normal hard-coded logic.
- (8) Faults should be confirmed using independent data when possible.

- (9) Recovery from a fault condition should result in a safe, unambiguous spacecraft state.
- (10) Audit trail design should be included early in the software design.
- (11) Fault protection routines should not interfere with planned or on-going spacecraft sequences.
- (12) The capability to rollback and resume critical sequences after fault correction activities should be considered in the design if transparency cannot be assured.
- (13) Actions taken under autonomous control, such as switching redundant elements, should be reversible.
- (14) Processors should perform a self-test to ensure proper operation before issuing fault correction commands.
- (15) An approach to testing and validating any routine should be developed as the routine is being designed. If test planning is deferred, the result is likely to be incomplete validation and increased risk.
- (16) Communications between processors should include extensive software checks. Examples include:
 - o Echoes
 - o Handshakes
 - o Repeated commands
 - o Parity checks
- (17) Software design checks should prohibit exceeding limits, entering forbidden zones, executing conflicting commands and similar faux pax. Examples include:
 - o Reasonableness checks of bounds, timing, overflow, etc.
 - o Constraint checks (e.g., of write protected areas).
 - o Mode conflict checks (e.g., of payload restrictions during maneuvers).
- (18) Sensors should be optimized to support on-board functions as well as traditional ground functions.

- (19) Computer memory margins should be preserved. (A 50% margin may not be unreasonable early in the design.) Memory margins can save money and reduce risk as development progresses. Software becomes the only tool available for changing system operation after launch.
- (20) Spacecraft system level testing should be designed from the outset to validate autonomous operations. Support equipment should be designed to permit visibility and control of autonomous operations.
- (21) Standby elements should be accessible for checkout, calibration and reprogramming in flight.
- (22) Fault protection should be applied at the lowest possible level. (Conversely, avoid involving the entire system if possible).
- (23) If fault protection is implemented by a central computer, it must be the most reliable element on the spacecraft.
- (24) Power switching should result in preservation of power margins and should not result in violation of thermal control requirements. The latter may impose time constraints on fault management actions.
- (25) Processors should protect against errors of commission and omission. (Voyager used processors acting in parallel and tandem, respectively, to achieve this.)
- (26) The spacecraft must protect itself against wrong or invalid ground commands.
- (27) Ground control must be able to override or disable autonomous software.
- (28) Thresholds for triggering fault routines should be set loosely until operating experience is accumulated.
- (29) Fault inputs (i.e., sensor signals) should be filtered to avoid reacting to transients.

APPENDIX A

VIKING ORBITER AND VOYAGER SPACECRAFT
DESIGN SUMMARIES

APPENDIX A

VIKING ORBITER AND VOYAGER SPACECRAFT DESIGN SUMMARIES

SECTION 1

VIKING ORBITER SPACECRAFT DESCRIPTION

The Viking orbiter spacecraft design goals were to support several phases of a 510 day prime mission. A Viking lander spacecraft would be supported during a 370 day cruise from Earth to Mars Orbit Injection (MOI). After the orbiter propulsion system performed MOI, a Visual Imaging Subsystem (VIS) would support a reconnaissance of the proposed landing sites. After lander separation and injection into transfer trajectory, the orbiter acted as a radio relay link between the lander and Earth. The VIS and two other science payload instruments would continue to provide orbital surveillance of the Martian surface for at least 90 days after the lander was down.

Two Viking spacecraft, each consisting of an orbiter and a lander, were launched in 1975. The orbiters were designated VO-1 and VO-2. After completion of the 510 day prime missions, both orbiters remained operational and entered extended mission phases which lasted until attitude control gas was exhausted. VO-2 achieved a total operating life of 1051 days, with VO-1 surviving for a total of 1785 days.

1.1 STRUCTURE AND CONFIGURATION

The Viking orbiter structure is based on a Mariner IX design, enlarged and modified to fit launch vehicle and lander interface requirements. The basic bus structure is an octagonal ring with alternating sides of lengths 20 inches and 55 inches. The height of the bus structure is 18 inches. Sixteen modular compartments provide space for equipment and thermal control louvers. The four solar panels extend on the $\pm x$ and $\pm y$ axes through the short (20 inch) sides and extend 32 feet from the tip of one panel to the tip of the opposite panel. Height of the orbiter is 10.4 feet including the propulsion system, but excluding the lander capsule.

The propulsion subsystem, scan platform, solar panels, and antennas are mounted external to the bus structure. Orbiter weight is 5,125 pounds with propellant.

Figure A-1 depicts the Viking orbiter spacecraft in its orbital configuration. A block diagram of the spacecraft subsystem is included in Figure A-2.

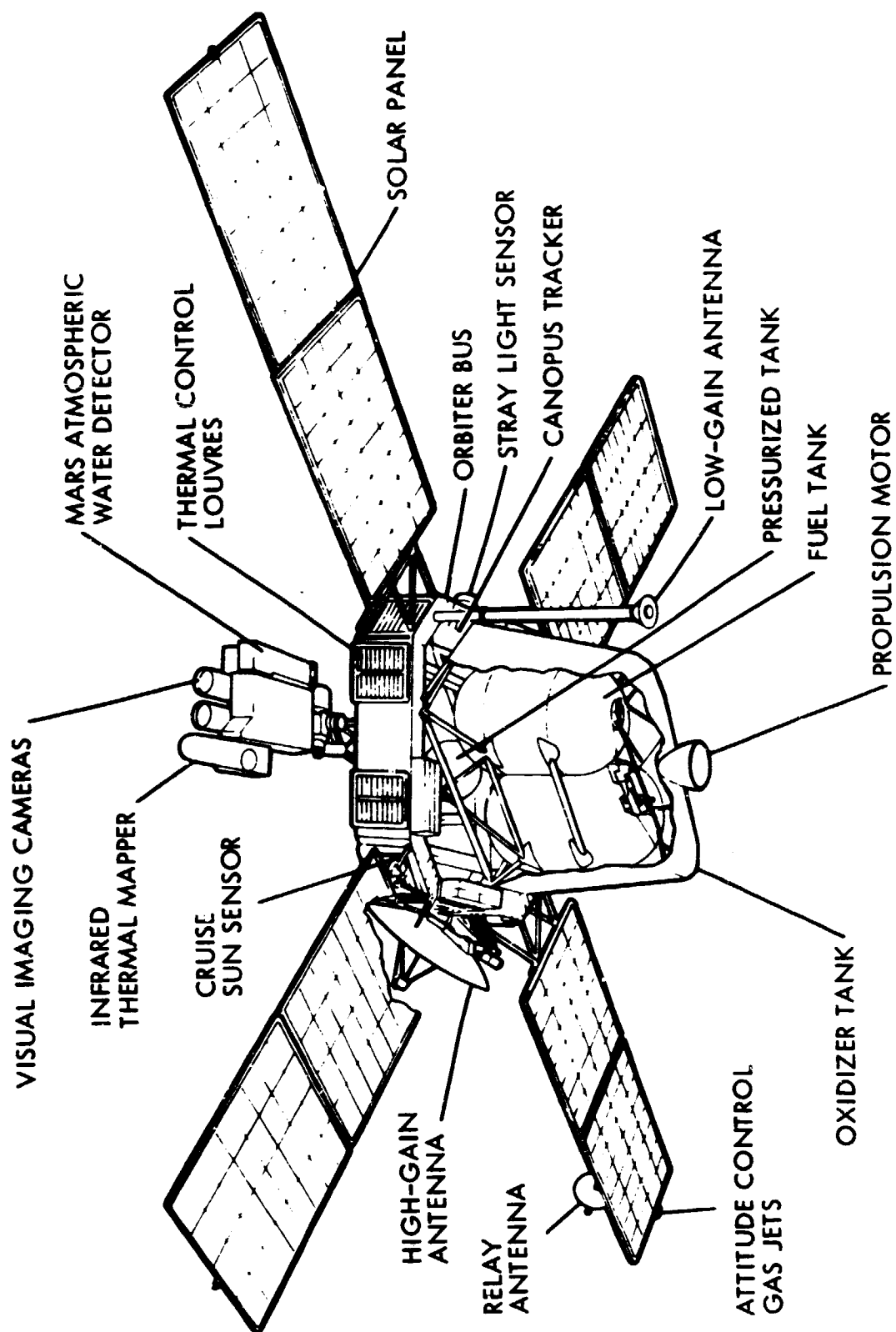


Figure A1-1. Viking Orbiter Configuration

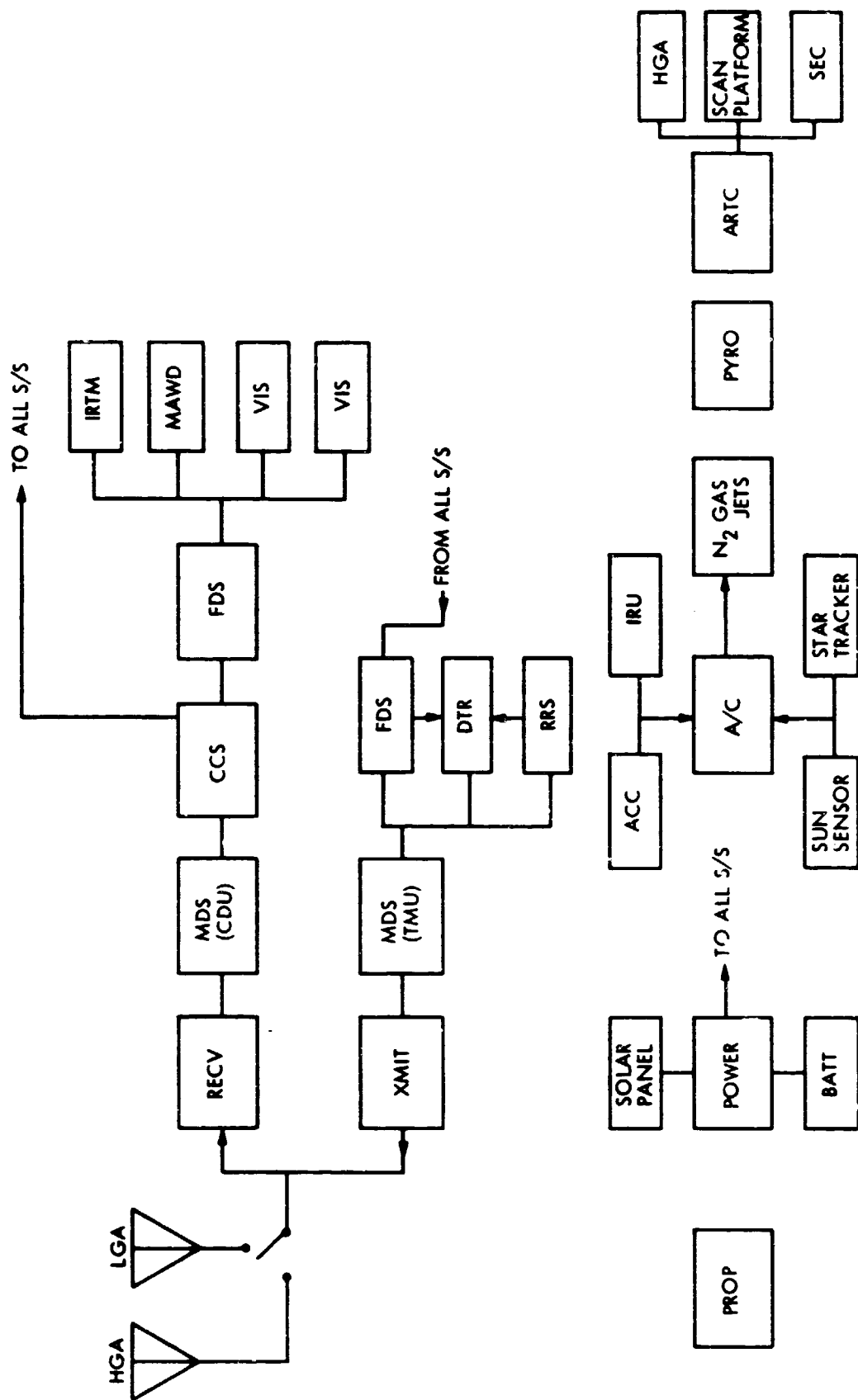


Figure A1-2. Viking Orbiter Block Diagram

1.2 COMMUNICATIONS SUBSYSTEM

The orbiter provides S-Band uplink and S-Band and X-Band downlinks for communications with earth. A Relay Radio Subsystem (RRS) provides a link at 382 megahertz to receive signals from a Viking lander on the Martian surface.

A 58 inch high gain parabolic antenna is mounted on the side of the orbiter between the +x and +y axes and gimballed with 2 degrees of freedom. S-Band signals are received at 2115 megahertz and transmitted at 2295 megahertz. The X-Band downlink is transmitted via the high gain antenna at 8455 megahertz. A separate low gain antenna with a wider coverage pattern can be used for S-Band uplink and downlink when the high gain antenna is not pointed at Earth.

The RRS antenna is mounted on the outer end of the +x solar panel. The lander data is passed to the recorders of the Data Storage Subsystem (DSS) and may be routed directly to the Modulation/Demodulation Subsystem (MDS) for real time transmission to Earth.

The MDS receives low rate engineering data and high rate science and engineering data from the Flight Data Subsystem (FDS) as well as direct data from the RRS and the lander prior to its separation. The DSS may also be played back to the MDS. The MDS then drives the Radio Frequency Subsystem (RFS), which transmits the downlink at X-Band and/or S-Band.

1.3 POWER SUBSYSTEM

The primary power source is the four solar panels with a total area of 23,250 square inches. These supply a total of 620 watts at the orbit of Mars. Two nickel-cadmium batteries back up the solar panels at peak power loads above 620 watts and during eclipse periods. Each battery has a 30 ampere-hour capacity at 57.4 volts charge. The batteries may be operated separately. All power subsystem commands are received from the Computer Command Subsystem (CCS).

1.4 COMPUTER COMMAND SUBSYSTEM

The CCS functions were to decode ground commands, issue commands to orbiter subsystems, store and execute sequences, and respond to spacecraft initiated interrupts. The CCS consists of two redundant computers with redundant memories, weighing a total of 42 pounds. Figure A-3 is a block diagram of the CCS. Each plated wire memory consists of 4096 words of 18 bits each. One-half of each memory is write protected to preserve control software and data constants. The other half of each memory may be erased and reloaded with software or data in flight. The computers are interrupt driven and may be operated in parallel or separately. The computers can also communicate with each other and have access to data in the engineering telemetry stream. External sources communicate with the CCS through bi-level inputs or binary word inputs. The CCS accepts 32 separate bi-level and 32 8-bit binary words.

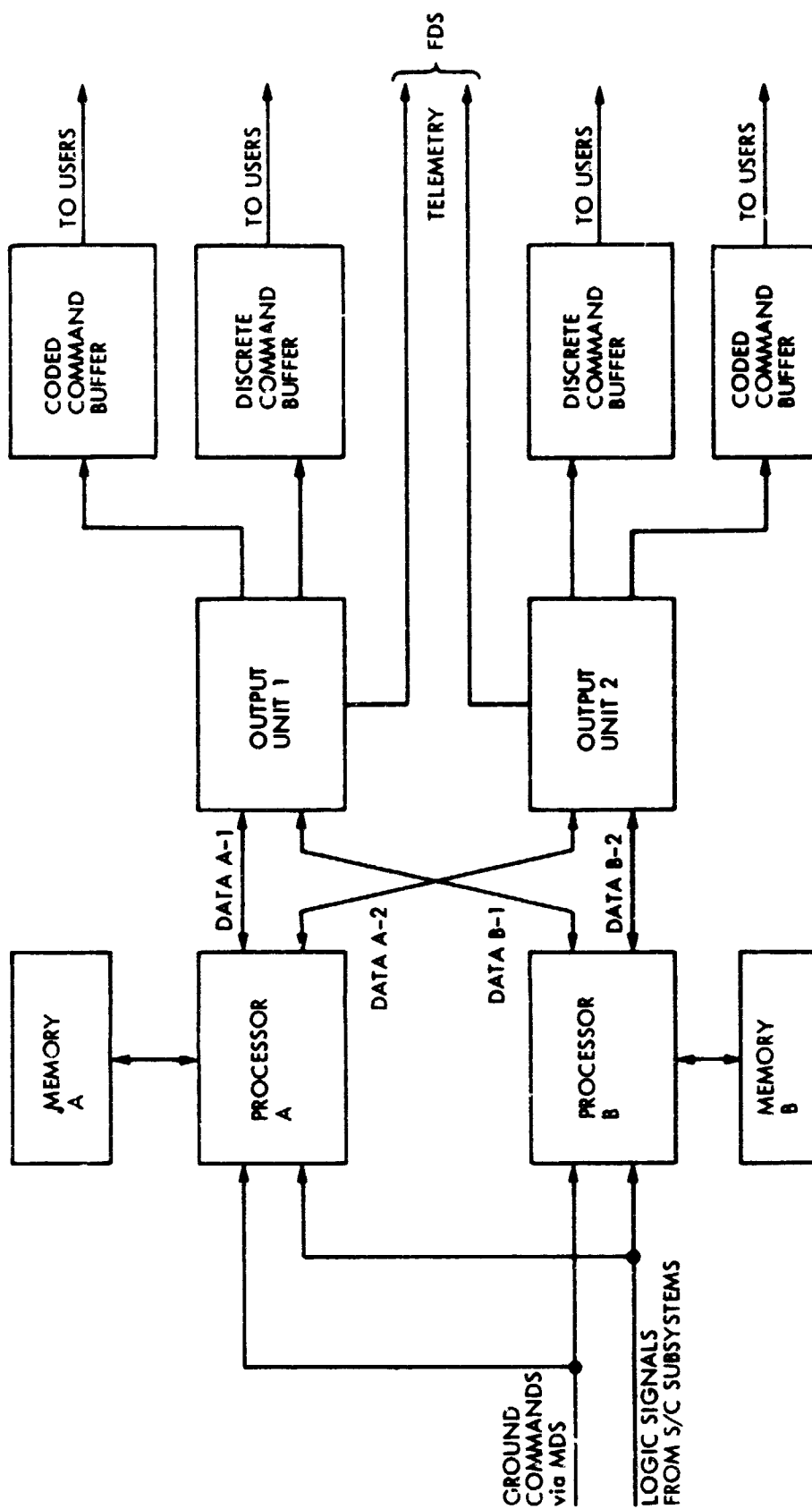


Figure A1-3. Viking Orbiter Computer Command Subsystem Block Diagram

1.5 PROPULSION SUBSYSTEM

The orbiter propulsion subsystem consists of a bipropellant rocket engine providing 300 pounds of thrust. It provides velocity change capability for cruise phase course corrections, Mars Orbit Insertion (MOI) burn, and orbit adjustments at Mars. The two propellant tanks are pressurized from a helium supply that can be isolated from the propellant tanks by a ladder of pyrotechnically actuated valves. The tanks are repressurized a maximum of three times in the mission. The helium supply has a valve allowing it to be used by the cold gas attitude control jets as a functionally redundant source. Total propellant capacity is 3,137 pounds.

The propulsion subsystem was sized to provide for four course correction burns, a nominal orbit insertion burn, and up to 20 nominal orbit trim burns. Burns are initiated and terminated by commands from the CCS. The attitude control system provides pitch and yaw steering commands during the burn and can move the thrust vector 9 degrees in either direction. Attitude control jets on the solar panels provide roll control under attitude control subsystem (ACS) command.

1.6 ATTITUDE CONTROL SUBSYSTEM

The ACS provides attitude control during all mission phases and propulsion burns. It additionally provides attitude reference data for navigation via optical sensing with the VIS. The ACS utilizes 4 sun acquisition sensors (one on each solar panel), a cruise sun sensor, and a Canopus star sensor. An inertial reference unit with two redundant sets of three single axis gyros and an accelerometer provides a functionally redundant capability for maneuvers and periods in Mars orbit when the sun is not visible.

An attitude control jet system supplied by pressurized nitrogen is utilized to provide control torques. The jets are positioned on the tips of the solar panels for maximum control authority. Pitch valves are on two opposite solar panels, roll and yaw jets are on the other two. ACS configuration and mode commands are received from the CCS.

There are two basic attitude control modes. The celestial mode utilizes the cruise sun sensor for pitch and yaw control and the Canopus star sensor for roll control. The Canopus sensor also provides an intensity signal to insure that it is locked on the proper star. The inertial reference unit provides three axis control for propulsion burns when celestial references are not in view. It also provides a control reference mode during solar eclipse periods in Mars orbit and at any time when the orbiter must be aligned to an attitude where the sun or Canopus are not visible from their respective sensors.

1.7 FLIGHT DATA SUBSYSTEM

The FDS collects orbiter/lander engineering data and science data, converts analog data to digital form, and forwards it to either the DSS or the communications subsystem for transmission. It also provides a timing reference to the orbiter scientific payload. The subsystem weighs 35 pounds and contains two 1024 word plated wire memories. An FDS word is 8 bits long. Three data transfer rates of 8.3, 33.3, and 2,000 bps are available.

1.8 DATA STORAGE SUBSYSTEM

The DSS consists of two independent and identical digital tape recorders. Each is an eight track reel-to-reel recorder capable of storing over 1/2 billion bits. This represents 55 Video Imaging System (VIS) picture frames per recorder. Five speeds of 1K, 2K, 4K, 8K, or 16K bits per second are available. Seven tracks can be recorded simultaneously at up to 301,212 bits per second. The eighth track will record at 4K or 16K bps.

1.9 TEMPERATURE CONTROL SUBSYSTEM

Temperature control is provided by multilayer insulation blankets and thermally operated louvers. Additional heat is provided to selected locations by electrical heaters.

1.10 AUTONOMOUS OPERATION

The CCS served as the spacecraft central processor and provided both sequence control and fault protection software. Fault routines stored in the CCS were designed to ensure completion of mission critical events such as MOI, maintain a command link in the event of a receiver failure or inability to process commands, manage critical power functions, and maintain Sun acquisition and downlink. Except for critical mission phases, such as MOI, the spacecraft usually reverted to a safe, commandable state after executing a fault protection algorithm.

Following the primary mission, an extended mission was conducted until both orbiters ran out of attitude control gas. The operations support staff was greatly reduced (from 650 people at the peak of the prime mission to 27 during the extended mission). Tracking time was very limited. In addition, the spacecraft had exceeded their design lifetime and various wearout and end-of-life phenomena began appearing. Operation of the CCS was changed to allow non-redundant execution, effectively increasing the capability of the on-board computer. The CCS was then reprogrammed with additional operational and fault protection routines. Access to the telemetry stream was available via the FDS-CCS interface and was used to extract engineering data for fault routines. Manpower intensive functions including battery charge control and attitude control gas jet leak clearing were performed by the spacecraft to reduce ground requirements.

Not all of the faults or conditions for which autonomous routines were provided were encountered during the prime and extended missions. Those

routines which were triggered generally performed as required and the experience gained was applied to other on-board problems. As a result, the prime mission design life of 510 days was exceeded with V0-2 operating for 1051 days and V0-1 for 1785 days. Both spacecraft were turned off when the attitude control gas was exhausted.

SECTION 2

VOYAGER SPACECRAFT DESCRIPTION

The Voyager mission consisted of two spacecraft, launched in 1977, which flew by Jupiter in 1979 and Saturn in 1980 and 1981. After Saturn encounter the second spacecraft, Voyager 2, will continue to Uranus, arriving in 1986. Voyager 1 is travelling out of the ecliptic.

Each spacecraft at launch consisted of a mission module - the planetary vehicle - and a propulsion module, which provided the final energy increment to inject the mission module onto the Jupiter trajectory. The propulsion module was jettisoned after the required velocity was attained. (For the major part of the mission, "spacecraft" and "mission module" are used interchangeably. In describing the prelaunch configuration and launch phase, "spacecraft" refers to the combined "mission module" and "propulsion module.")

The mission module after injection weighed 825 kilograms (1,819 pounds) including a 117-kg (258-lb.) science instrument payload. The propulsion module, with its large solid-propellant rocket motor, weighed 1,207 kg (2,660 lb.). Launch weight of the spacecraft was 2,066 kg (4,555 lb.).

During cruise, the spacecraft is three-axis stabilized using the Sun and a star (usually Canopus) as celestial references. Hydrazine (mono-propellant) jets provide thrust for attitude stabilization and for Trajectory Correction Maneuvers (TCM).

Three engineering subsystems are programmable for onboard control of most spacecraft functions. Only trajectory-correction maneuvers must be enabled by ground command. The three subsystems are the Computer Command Subsystem (CCS), Flight Data Subsystem (FDS) and Attitude and Articulation Control Subsystem (AACS). The memories of the units can be updated or modified by ground command.

A nuclear power source, three Radioisotope Thermoelectric Generators (RTG), provides electrical power for the spacecraft.

The pointable science instruments are mounted on a commandable (two-axis) scan platform at the end of the science boom for precise pointing. Other body-fixed and boom-mounted instruments are aligned to provide for proper interpretation of their measurements.

Data storage capacity on the spacecraft is about 536 million bits - approximately the equivalent of 100 full-resolution photos.

Dual-frequency communications links, S-Band and X-band, provide accurate navigation data and large amounts of science information during planetary encounter periods (up to 115,200 bits per second at Jupiter and 44,800 bps at Saturn).

Figure A-4 depicts the configuration of the Voyager spacecraft.

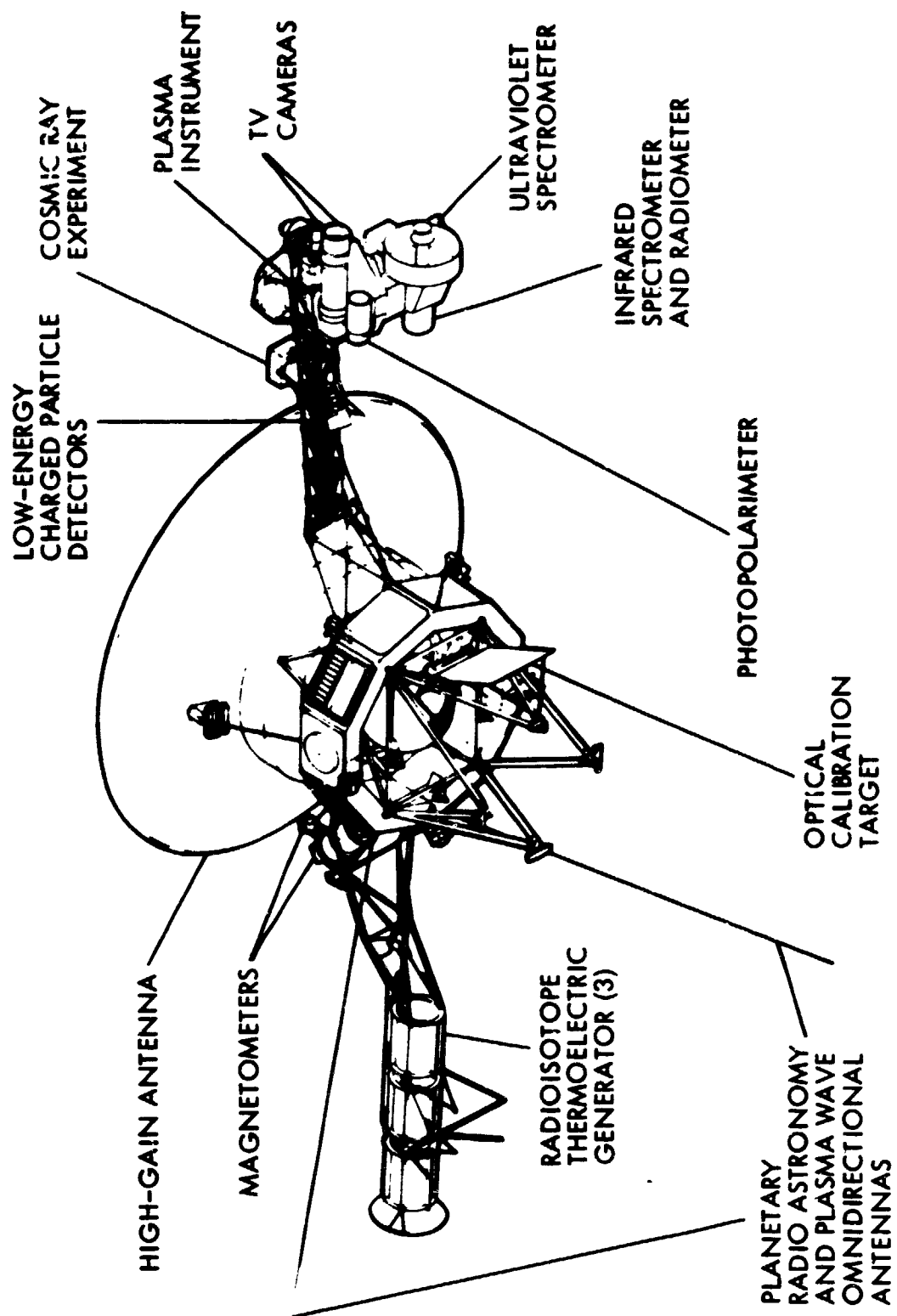


Figure A1-4. Voyager Spacecraft Configuration

2.1 STRUCTURE AND CONFIGURATION

The basic mission module structure is a 24.5-kg (54-lb.) 10-sided aluminum framework with 10 electronics packaging compartments. The structure is 47 centimeters (18.5 in.) high, 178 cm (70 in.) across from flat to flat, and 188 cm (74 in.) from longeron to opposite longeron. The electronics assemblies are structural elements of the 10-sided box.

The spherical propellant tank that contains hydrazine fuel for attitude control thrusters and trajectory correction maneuvers occupies the center cavity of the decagon. Propellant lines carry hydrazine to 12 small attitude-control and four trajectory correction maneuver thrusters on the mission modules and to larger thrust-vector-control engines on the propulsion module during launch.

The 3.66 m (12 ft.) diameter high-gain parabolic reflector is supported above the basic structure by a tubular truss work. The antenna reflector has an aluminum honeycomb core and is surfaced on both sides by graphite epoxy laminate skins. Attachment to the trusses is along a 178-cm (70 in.) diameter support ring. The Sun sensor protrudes through a cutout in the antenna dish. An X-Band feed horn is at the center of the reflector. Two S-Band feed horns are mounted back-to-back with the X-Band subreflector on a three-legged truss above the dish. One of the S-Band feed horns functions as the low-gain antenna.

Louver assemblies for temperature control are fastened to the outer faces of two electronics compartments - those housing the power conditioning assembly and the radio transmitter power amplifiers. The top and bottom of the 10-sided structure are enclosed with multi-layer thermal blankets.

Two Canopus star tracker units are mounted side-by-side and parallel atop the upper ring of the decagon.

Three RTG's are assembled in tandem on a deployable boom hinged on an outrigger arrangement of struts attached to the basic structure. Each RTG unit weighs 39 kg (86 lb). Together, they provide over 400 watts of electrical power to the spacecraft.

The science boom, supporting the instruments most sensitive to radiation, is located 180 degrees from the RTG boom. The two-axis scan platform is mounted at the end of the boom and provides precision pointing for four remote-sensing instruments, the ultraviolet spectrometer, infrared interferometer spectrometer and radiometer, photopolarimeter (no longer operating on Voyager 1), and a two-camera imaging science subsystem. Total platform gimballed weight is 103 kg (227 lb).

The magnetic fields experiment consists of an electronics subassembly located in one of the mission module electronics bays and four magnetometers - two high-field sensors affixed to the spacecraft and two low-field sensors mounted on a 13-m (43 ft.) deployable boom. The boom, constructed of epoxy glass, spiralled from its stowed position in an aluminum cylinder to form a rigid triangular mast with one magnetometer attached to its end plate and another positioned 6 m (19.6 ft.) closer to the spacecraft. The mast weighs 2.26 kg (5 lb.), a few ounces less than the cabling running its length and carrying power to and data from the magnetometers. The boom housing is a 22.8 cm (9 in.) diameter cylinder, 66 cm (26 in.) long, supported by the RTG outrigger. The mast uncoils in helical fashion along a line between the rear face of the high-gain antenna and the RTG.

A 0.36 square m (4 sq. ft.) shunt radiator/science calibration target faces outward from the propulsion module truss adapter toward the scan platform. The dual-purpose structure is a flat sandwich of two aluminum radiating surfaces lining a honeycomb core. Through resistors between the plates, excess electrical power from the RTG can be radiated to space as heat. The outer surface also serves as a photometric calibration target for the remote-sensing science instruments on the scan platform.

2.2 COMMUNICATIONS SUBSYSTEM

Communications with the Voyagers is by radio link between Earth tracking stations and a dual-frequency radio system aboard the spacecraft. The uplink operates at S-Band only, carrying commands and ranging signals from ground stations to one of a pair of redundant receivers. The downlink is transmitted from the spacecraft at both S-Band and X-Band frequencies.

The on-board communications subsystem also includes a programmable Flight Data Subsystem, Modulation/Demodulation Subsystem, Data Storage Subsystem (DSS) and High-Gain and Low-Gain Antennae.

The FDS, one of the three on-board computers, controls the science instruments and formats all science and engineering data for telemetry to Earth. The Telemetry Modulation Unit of the MDS feeds data to the downlink. The Command Detector Unit of the MDS detects ground commands received by the spacecraft and routes them to the CCS for decoding.

Only one receiver is powered at any one time, with the redundant receiver on standby. The receiver operates continuously during the mission at

about 2113 megahertz. Different frequency ranges have been assigned to the Radio-Frequency Subsystem of each spacecraft. The receiver can be used with either the High-Gain (dish) or Low-Gain (omni) Antenna. (Voyager 2's primary receiver failed on April 5, 1978, and the spacecraft is operating on its back-up receiver.)

The S-Band transmitter consists of two redundant exciters and two redundant RF power amplifiers, of which any combination is possible. Only one exciter-amplifier combination operates at any one time. Selection of the combination is by on-board failure-detection logic within the Computer Command Subsystem, with ground-command backup. The same arrangement of exciter-amplifier combinations makes up the X-Band transmitting unit.

One S-Band and both X-Band amplifiers employ Traveling Wave Tubes. The second S-Band unit is a solid state amplifier. The S-Band transmitter is capable of operating at 9.4 watts or at 28.3 watts when switched to high power and it can radiate from both antennas. X-Band power output is 12 watts and 21.3 watts. X-Band uses only the high-gain antenna. (S-Band and X-Band never operate at high power simultaneously due to raw power availability and thermal control considerations).

When no uplink signal is being received, the transmitted S-Band frequency of about 2295 MHz and X-Band frequency of 8418 MHz originate in the S-Band exciter's auxiliary oscillator or in a separate Ultra-Stable Oscillator (one-way tracking). With the receiver phase-locked to an uplink signal, the receiver provides the frequency source for both transmitters (two-way tracking). The radio system can also operate with the receiver locked to an uplink signal while the downlink carrier frequencies are determined by one of the on-board oscillators (two-way, noncoherent tracking).

The X-Band downlink was not normally used during the first 80 days of the mission, until Earth was within the beam of the spacecraft's high-gain antenna. Communications during launch, near-Earth and early cruise phase operations were confined to S-Band and the low-gain antenna. An exception occurred early in the flight when the spacecraft, on inertial control, pointed the high-gain antenna toward Earth to support instrument calibration and an optical navigation/high-rate telecommunications link test. During its calibration sequence on Sept. 18, 1977, Voyager 1 took pictures of the Earth-Moon system.

Under normal conditions, after the first 80 days of the mission, all communications, both S-Band and X-Band, have been via the high-gain antenna. X-Band is turned off, however, and the S-Band transmitter and receiver are switched to the low-gain antenna during periodic science maneuvers and trajectory correction maneuvers.

The S-Band downlink is always on, operating at high power during spacecraft maneuvers or during the cruise phase only when the 26 m antenna Deep Space Network stations are tracking and at low power whenever X-Band is on. At Saturn, both S-Band and X-Band transmitters will be at low power when gyros and tape recorder are on simultaneously.

2.2.1 Commanding the Spacecraft

Ground commands are used to put into operation selected flight sequences or to cope with unexpected events. Commands are issued in either a predetermined, timed sequence via on-board program control or directly as received from the ground. Most commands are issued by the spacecraft's CCS in its role as "sequencer-of-events" and by the Flight Data Subsystem as controller of the science instruments because the time delays associated with the extreme distances for command signals to reach the spacecraft eliminate the usual "real-time" control procedures.

All communications between spacecraft and Earth are in digital form. Command signals, transmitted at 16 bits per second to the spacecraft, are detected in the Command Detector Unit and routed to the Computer Command Subsystem for further routing to their proper destination. Ground commands to the spacecraft fall into two major categories: discrete commands, and coded commands.

A discrete command causes a single action on the spacecraft. For example, DC-2D switches the S-Band amplifier to high power; DC-2DR, switches the S-Band amplifier to low power; DC-2E, S-Band radiates from high-gain antenna; DC-2ER, S-Band transmits from the low gain antenna. Coded commands are the transfer of digital data from the Computer Command Subsystem or directly from the ground via the Computer Command Subsystem to user subsystems. Subsystems receiving coded commands are Flight Data, Attitude and Articulation Control, Modulation/Demodulation, Data Storage and Power.

Ground commands back up all critical spacecraft functions that, in a standard mission, are initiated automatically by on-board logic. Command modulation will be off during science maneuvers and trajectory correction maneuvers unless a spacecraft emergency arises.

2.2.2 Downlink Telemetry

Data telemetered from the spacecraft consists of engineering and science measurements prepared for transmission by the FDS, Telemetry Modulation Unit and Data Storage Subsystem. The encoded information will indicate voltages, pressures, temperatures, television pictures and other values measured by the spacecraft telemetry sensors and science instruments.

Two telemetry channels, low rate and high rate, are provided for the transmission of spacecraft data. The low rate channel functions only at S-Band at a single 40 bps data rate and contains real time engineering data exclusively. It is on only during planetary encounters when the high rate channel is operating at X-Band.

The high-rate channel is on throughout the mission and operates at either S-Band or X-Band and contains the following types of data:

- (1) Engineering only at 40 bps or 1,200 bps transmitted at S-Band only. The higher data rate usually occurs only during launch and trajectory correction maneuvers.
- (2) Real-time cruise science and engineering at 2,560, 1,280, 640, 320, 160, and 80 bps transmitted at S-Band only. 40, 20 and 10 bps may be used for post-Saturn operations.
- (3) Real-time encounter general science and engineering at 7.2 kilobits per second (kbps). A special 115.2 kbps rate (available at Jupiter for the planetary radio astronomy and plasma wave instruments) was transmitted at X-Band only.
- (4) Real-time encounter general science, engineering, and television at 115.2, 89.6, 67.2, 44.8, 29.86, and 19.2 kbps transmitted at X-Band only.
- (5) Real-time encounter general science and engineering, plus tape recorder playback, at 67.2 and 44.8 kbps and 29.86 kbps transmitted at X-Band only.
- (6) Recorded data playback only at 21.6 and 7.2 kbps transmitted at X-Band only.
- (7) Memory data stored in the three on-board computers, CCS, FDS, and AACS, are read out and played back at 40 or 1,200 bps, and transmitted at either S-Band or X-Band (treated as engineering data).

The many data rates for each type of telemetered information are required by the changing length of the telecommunications link to Earth and possible adverse effects of Earth weather upon reception of X-Band radio signals. The S-Band cruise science primary telemetry rate is 2,560 bps. Lesser rates result in reduced instrument sampling and will be used only when the telecommunications link cannot support the higher rate.

In order to allow real-time transmission of video information at each encounter, the FDS can handle the imaging data at six downlink rates from 115.2 to 19.2 kbps. The 115.2 kbps rate represents the standard full-frame readout (at 48 seconds per frame) of the TV vidicon. Under normal conditions, that rate was used at Jupiter. Full-frame, full-resolution TV from Saturn can be obtained by increasing the frame readout time to 144 seconds (3:1 slow scan) and transmitting the data at 44.8 kbps. A number of other slow scan and frame-edit options are available to match the capability of the telecommunications link.

2.2.3 Tracking the Spacecraft

Very precise navigation is required to achieve the desired maneuver and flyby accuracies for a multi-planet/satellite encounter mission.

To provide Doppler tracking data, the S-Band signal transmitted from Earth is received at the spacecraft, changed in frequency by a known ratio and retransmitted to Earth. It is possible to precisely determine the transmitted downlink frequency while measuring the Doppler-shifted received signal, thereby measuring spacecraft velocity. This is called coherent two-way tracking.

One-way tracking is achieved when no uplink signal is received and the downlink carrier frequency is provided by an on-board oscillator. Noncoherent two-way tracking occurs when uplink and downlink carriers are operating independently.

When both S-Band and X-Band transmitters are on, X-Band frequency will always be eleven-thirds the S-Band frequency regardless of the frequency source - spacecraft receiver, Ultra-Stable Oscillator or S-Band exciter auxiliary oscillator.

Distance or range to the spacecraft is measured in the coherent two-way configuration by transmitting a digital code (ranging modulation) on the uplink, turning this code around in the spacecraft and sending it back to the ground. By measuring the total elapsed time between transmitting and receiving the code at the ground station, and knowing such factors as the speed of light, turnaround delay, and the relative velocity of the spacecraft with respect to the tracking station, it is possible to determine spacecraft range.

Dual-frequency ranging with both S-Band and X-Band ranging on is conducted during planetary phases of the mission and during the cruise phases when the Deep Space Network's 64 m (210 ft.) antennas are tracking. Special three-way dual-frequency ranging cycles will be conducted while two or more ground stations on two continents are tracking the spacecraft.

All ranging modulation is turned off during science maneuvers, trajectory correction maneuvers and planetary occultations.

2.3 POWER SUBSYSTEM

The Voyager power subsystem supplies all electrical power to the spacecraft by generating, converting, conditioning and switching the power.

The power source for the mission module is an array of three Radioisotope Thermoelectric Generators (RTG). The propulsion module, active only during the brief injection phase of the mission, used a separate battery power source.

The RTG units, mounted in tandem on the deployable boom and connected in parallel, convert to electricity the heat released by the

isotopic decay of Plutonium 238. Each isotope heat source has a capacity of 2,400 watts thermal with a resultant maximum power output of 160 watts at the beginning of the mission. There is a gradual decrease in power output with time. The minimum total power available from the three RTGs on each Voyager ranges from about 450 watts within a few hours after launch to about 430 watts after the spacecraft passes Saturn.

Spacecraft power requirements from launch to post-Saturn operation are characterized by this general power timeline; launch and post-launch, 235 to 265 watts; interplanetary cruise, 320 to 365 watts; Jupiter encounter, 384 to 401 watts; Saturn encounter, 377 to 382 watts; and post-Saturn, less than 365 watts.

Telemetry measurements have been selected to provide the necessary information for power management by ground command, if needed.

Power from the RTG's is held at a constant 30 volts DC by a shunt regulator. The 30 volts are supplied directly to some spacecraft equipment and are switched to others in the power distribution subassembly. The main power inverter also is supplied the 30 volts DC for conversion to 2.4 kHz square wave AC used by some spacecraft subsystems. Again, the AC power may be supplied directly to equipment or can be switched on or off by power relays.

Command-actuated relays control the distribution of power in the spacecraft. Some relays function as simple on-off switches and others transfer power from one module to another within a subsystem.

Among the users of DC power, in addition to the inverter, are the radio subsystem, gyros, propulsion isolation valves, some science instruments, most temperature control heaters and the motors that deployed the planetary radio astronomy antennas. Other elements of the spacecraft use the AC power.

There are two identical 2.4 kHz power inverters - main and standby. The main inverter is on from launch and remains on throughout the mission. In case of malfunction or failure in the main inverter, the power chain, after a 1.5-second delay, is switched automatically to the standby inverter. Once the switchover is made, it is irreversible.

A 4.8 kHz sync and timing signal from the FDS is used as a frequency reference in the inverter. The frequency is divided by two, and the output is 2.4 kHz. The AC regulator is accurate to .004 percent. The 4.8 kHz timing signal is sent, in turn, to the CCS, which contains the spacecraft's master clock.

Because of the long mission lifetime, charge capacitor energy-storage banks are used instead of batteries to supply the short-term extra power demanded by instantaneous overloads that would cause the main DC power voltage to dip below acceptable limits. A typical heavy transient overload occurs at turn-on of a radio power amplifier.

Full output of the RTGs, a constant power source, must be used or dissipated in some way to prevent overheating of the generator units and DC

voltage rising above the allowed maximum. That is controlled by a shunt regulator that consumes excess RTG output power above that required to operate the spacecraft. The excess power is dissipated as heat in resistors in the shunt radiator mounted outside the spacecraft and radiated into space.

2.4 COMPUTER COMMAND SUBSYSTEM

The heart of the on-board control system is the CCS, which includes two independent memories, each with a capacity of 4,096 data words. Half of each memory stores reusable fixed routines that do not change during the mission. The second half is programmable by updates from the ground.

Most commands to other spacecraft subsystems are issued from the CCS memory, which, at any given time, is loaded with the sequences appropriate to the mission phase. The CCS also can decode commands from the ground and pass them along to other spacecraft subsystems.

Under control of an accurate on-board clock, the CCS counts hours, minutes or seconds until some preprogrammed interval has elapsed and then branches into subroutines stored in memory that result in commands to other subsystems. A sequencing event can be a single command or a routine that includes many commands (e.g., manipulating the tape recorder during a playback sequence).

The CCS can issue commands from one of its two processors or from both in parallel or tandem. An example of CCS dual control is the execution of Trajectory Correction Maneuvers. Trajectory Correction Maneuver thrusters are started with a tandem command (both processors must send consistent commands to a single output unit) and stopped with a parallel command (either processor working through different output units can stop the burn).

The Computer Command Subsystem can survive any single, internal fault; each functional unit has a duplicate elsewhere in the subsystem.

2.5 PROPULSION SUBSYSTEM

The propulsion subsystem consists of a large solid-propellant rocket motor for final Earth-to-Jupiter trajectory velocity increment and a hydrazine blowdown system that fuels 16 thrusters on the mission module.

The single hydrazine (N_2H_4) supply is contained under a 71-cm (28 in.) diameter spherical titanium tank, separated from the helium pressurization gas by a Teflon-filled rubber diaphragm. The tank is located in the central cavity of the mission module. It is surrounded by a structure that contains 104 kg (230 lb.) of hydrazine at 100°C and 100 atm. The tank is rated at 100 Newtons per square meter (2.25 lb./sq. in.) at 100°C and 100 atm.

helium pressure will decrease to a minimum of about 130 psi.

The 16 thrusters on the mission module each deliver 0.9 N (0.2 lb.) thrust. Four are used to execute trajectory correction maneuvers; the others, in two redundant six-thruster branches, are used to stabilize the spacecraft about its three axes. Only one branch of attitude control thrusters is needed at any time.

2.6 ATTITUDE AND ARTICULATION CONTROL SUBSYSTEM

The AACS includes an on-board computer called HYPACE (Hybrid Programmable Attitude Control Electronics), redundant sun sensors, redundant Canopus star trackers, three two-axis gyros, and scan actuators for positioning the science platform.

The HYPACE contains two redundant 4,096-word plated-wire memories, part of which are fixed and part programmable, plus redundant processors and input/output driver circuits. For a nominal mission, the memories will be changed only to modify predetermined control instructions.

2.6.1 Celestial Reference Control

The Sun sensors, which look through a slot in the High-Gain Antenna dish, are electro-optical devices that send attitude position error signals to HYPACE, which in turn, signals the appropriate attitude control thrusters to fire and turn the spacecraft in the proper direction. Sun lock stabilizes the spacecraft about two axes (pitch and yaw).

The star Canopus, one of the brightest in the galaxy, is usually the second celestial reference for three-axis stabilization. Two Canopus trackers are mounted so that their lines of sight are parallel. Only one is in use at any one time.

The star tracker, through HYPACE logic, causes the thruster to roll the spacecraft about the already-fixed Earth or Sun-pointed roll axis until the tracker is locked on Canopus. Brightness of the tracker's target star is telemetered to the ground to verify that the correct star has been acquired.

One of the Canopus star trackers on Voyager 1 exhibited degraded performance beginning in April, 1990. Extensive ground and flight testing has led to both a high-confidence failure model as well as complete characterization of the Canopus star tracker's degraded performance. Accordingly, it is planned to continue using this unit rather than switch to the backup unit.

To enhance downlink communication capability, the Sun sensor is also used to provide attitude information to the ground. The Sun sensor can be used to provide attitude information to the ground. The Sun sensor can be used to provide attitude information to the ground.

Three axis stabilization with celestial reference is the normal attitude control mode for cruise phases between planets.

2.6.2 Inertial Reference Control

The spacecraft can be stabilized about one axis (roll) or all three axes with the AACS Inertial Reference Unit consisting of three gyros.

Appropriate inertial reference modes are used whenever the spacecraft is not in Sun/star celestial lock. Such situations include: maintaining inertial reference from Centaur separation until initial celestial acquisition is achieved; purposely turning the spacecraft off Sun/star lock to do required trajectory corrections or science instrument mapping or calibrations; providing a reference when the Sun is occulted; and providing a reference when concern exists that the Canopus or Sun sensor will detect stray reflected light intensity from unwanted sources such as planets, rings, or satellites.

Each gyro has associated electronics to provide position information about two orthogonal axes (Gyro A: pitch and yaw; Gyro B: roll and pitch; Gyro C: yaw and roll). Normally, two gyros are on for any inertial mode. The gyros have two selectable rates: high rate for the propulsion module injection phase; and the other for mission module cruise and trajectory correction and science maneuvers.

2.6.3 Trajectory Correction Maneuvers

The Voyager trajectories are planned around nine Trajectory Correction Maneuvers for each spacecraft between launch and Saturn encounter. Mission requirements call for extremely accurate maneuvers to reach the desired aiming zones at Jupiter, Saturn and the target satellites. Total velocity increment capability for each spacecraft is about 190 meters per second (mps).

Trajectory Correction Maneuver sequencing is under control of the CCS, which sends the required turn angles to the AACS for positioning the spacecraft at the correct orientation in space and, at the proper time, sends commands to the AACS to start and stop the Trajectory Correction Maneuver burn. Attitude control is maintained by pulse-off sequencing of the Trajectory Correction Maneuver engines and pulse-on sequencing of two attitude-control roll thrusters. Position and rate signals are obtained from the gyros. After the burn, reacquisition of the cruise celestial references is accomplished by unwinding the commanded turns - repeating the turn sequence in reverse order. All Trajectory Correction Maneuvers are enabled by ground command.

2.6.4 Science Platform (Articulation Control)

Voyager's two television cameras, ultraviolet spectrometer, photopolarimeter, infrared spectrometer, and radiometer are mounted on the scan platform that can be rotated about two axes for precise pointing at

Jupiter, Saturn and their moons during the planetary phases of the flight. The platform is located at the end of the science boom. Total gimbaled weight is 102.5 kg (226 lb.).

Controlled by the AACS, the scan platform allows multiple pointing directions of the instruments. Driver circuits for scan actuators, one for each axis, are located in the AACS. The platform's two axes of rotation are described as azimuth angle motion about an axis displaced 7 degrees from the spacecraft roll axis (perpendicular to the science boom centerline) and elevation angle motion about an axis perpendicular to the azimuth axis and rotating with the azimuth axis. Angular range is 360 degrees in azimuth and 210 degrees in elevation.

The platform is slewed one axis at a time with selectable slew rates in response to CCS commands to the AACS. Slew rates are: 0.083 degrees per second; and a special Ultraviolet Spectrometer low rate: 0.0052 degrees per second. Camera line-of-sight is controlled to within 2.5 milliradians.

2.7 TEMPERATURE CONTROL

Both the top and bottom of the Mission Module's basic decagon structure are enclosed with multi-layer thermal blankets to prevent the rapid loss of heat to space. The blankets are sandwiches of aluminized Mylar, sheets of Teflon for micrometeoroid protection and outer black Kapton covers that are electrically conductive to prevent the accumulation of electrostatic charges.

Also extensively blanketed are the instruments on the scan platform. Smaller blankets and thermal wrap cover eight electronics bays, boom and body-mounted instruments, cabling, propellant lines and structural struts. Only a few exterior elements of the spacecraft are not clad in the black film - the High-Gain Antenna reflector, plasma sensors, sun sensors and antenna feed cones.

Temperature control of four of the 10 electronics compartments is provided by thermostatically controlled louver assemblies that provide an internal operating range near room temperature. The louvers are rotated open by bimetallic springs when large amounts of heat are dissipated. These bays contain the power-conditioning equipment and the radio power amplifiers. Mini-louvers are located on the scan platform, cosmic ray instrument and Sun sensors.

Radioisotope heating units are located in the magnetometer sensors and the Sun sensors. No radioisotope heating units are used near instruments that detect charged particles. Electric heaters are located throughout the spacecraft to provide additional heat during portions of the mission. Many of the heaters are turned off when their respective valves, instruments or subassemblies are on and dissipating power.

Three of the engineering subsystems are programmable and provide on-board control of most spacecraft functions. The Computer Command Subsystem (CCS) acts as the central executive, processes commands, and controls spacecraft sequencing. The Flight Data Subsystem (FDS) controls the science instruments and processes all engineering telemetry and science data. The Attitude and Articulation Control Subsystem (AACS) controls spacecraft attitude and scan platform pointing. Each computer is redundant, and each computer contains two redundant memories, each with 4,096 words, in the CCS and AACS, and 3,192 words in the FDS.

Fault protection software is contained in both the CCS and AACS, with the CCS serving as the spacecraft executive. Table A-1a lists the CCS routines and their sizes. CCS fault routines are initiated by external interrupts, followed by preprogrammed responses. All routines were used for both Voyager 1 and 2 and were resident in both CCS memories. Table A-1b lists the AACS routines and their sizes. AACS fault routines are executed periodically and compare current performance indicators against preprogrammed values. Corrective action is initiated when an unfavorable comparison occurs. Figure A1-5 shows the AACS software flow and processing intervals.

Both Voyagers successfully completed the Jupiter and Saturn encounters and are conducting extended missions. On-board software routines have generally performed as designed. Flight experience has indicated that adequate modeling and test of flight environments is critical to successful fault routine performance and that parameter levels used to trigger fault routines should not be set too tight to avoid unnecessary fault management actions.

Table A-1a. Voyager CCS Fault Protection Routines

Routine	Number of CCS Words	% of One CCS Memory
CCS Error	230	5.6
AACS Power Code Processing	339	8.3
Command Loss	101	2.5
IRIS Power	20	.5
Power Check	167	4.1
Radio Frequency Loss	93	2.3
Direct Memory Load	67	1.6
Turn Support	68	1.7
TOTAL	1085	26.5%

Table A-1b. Voyager AACS Fault Protection Routines

Routine	Number of AACS Words	% of One AACS Memory
Gyro Fault Protection	165	4.0
Thruster Failure	95	2.4
Power Supply Monitor	22	0.6
Plated Wire Refresh	27	0.6
Processor Test Control	19	0.4
CCS Command Interpreter	35	0.8
Celestial Sensor Logic	135	3.2
Power Code Processor	64	1.6
Catastrophe Handler	32	0.8
Miscellaneous Other Functions	200	4.8
TOTAL	794	19.4%

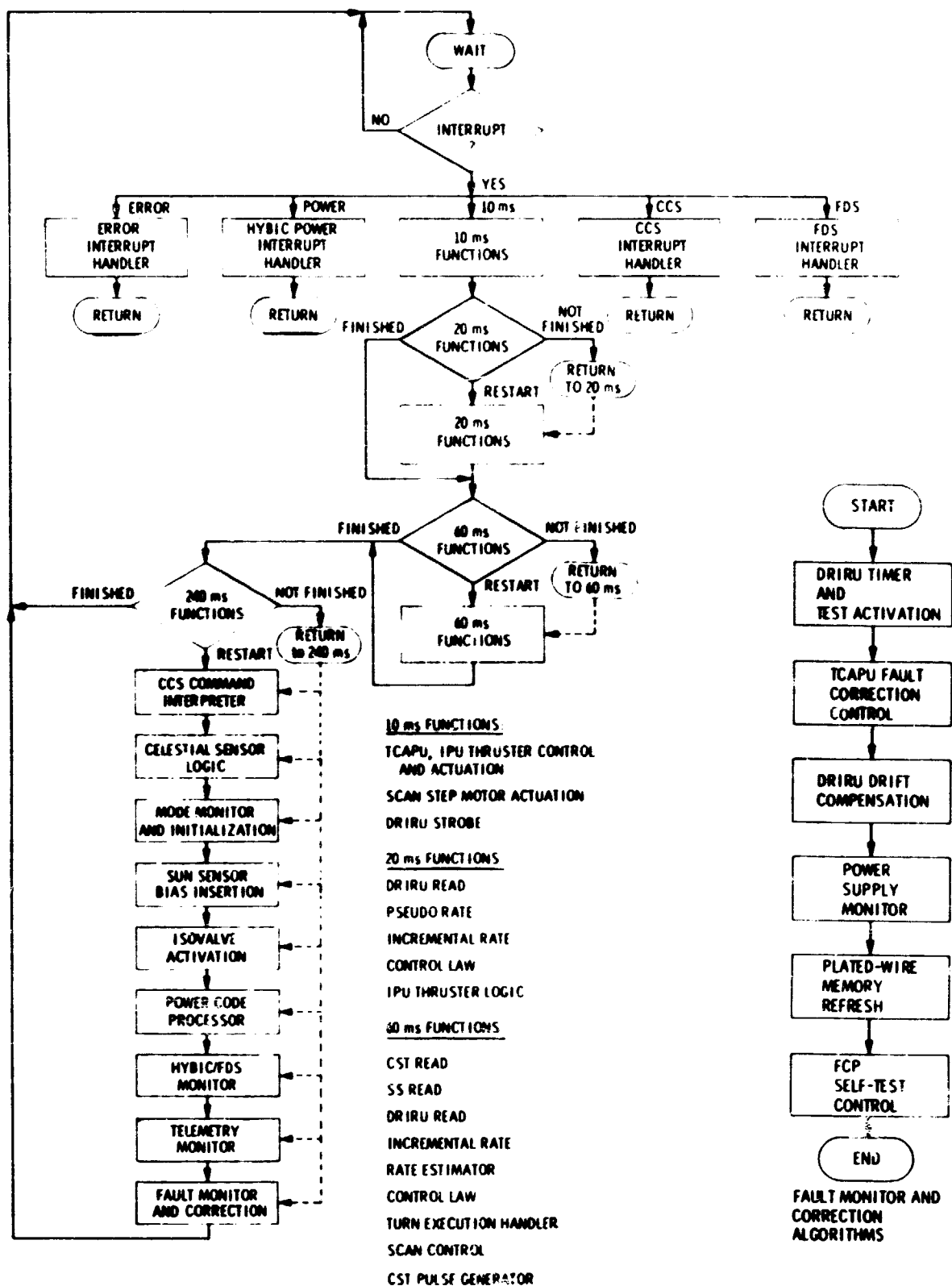


Figure A1-5. Voyager AACs Software Fault Monitor and Correction Algorithms

APPENDIX B
GENERALIZED ALGORITHMS FOR AUTONOMOUS CONTROL
AND FAULT MANAGEMENT

TABLE OF CONTENTS

APPENDIX B

	<u>Page No.</u>
SECTION B1: Loss of Attitude Reference & Reacquisition	B1-1
SECTION B2: Power Load Fault Management	B2-1
SECTION B3: Power Processor Fault Management	B3-1
SECTION B4: Battery Cell Replacement	B4-1

Appendix B

Section B1

Loss of Attitude References and Reacquisition

DESCRIPTION OF AN AUTONOMOUS FUNCTION

SECTION 1

FUNCTION NAME: LOSS OF ATTITUDE REFERENCES AND REACQUISITION

SECTION 2

FUNCTIONAL DESCRIPTION

An autonomous spacecraft is assumed to use celestial references to establish and maintain attitude control. Loss of such references should result in the spacecraft initiating sequences necessary for reference recovery.

An attitude control subsystem (ACS) uses signals provided by sensors to determine the presence of celestial references and to provide an estimate of the change in spacecraft position relative to these references. A notion of reference loss then may be defined in terms of these sensor signals.

2.1 FUNCTIONAL BLOCK DIAGRAMS

Figure B1-1 is a block diagram showing inputs and outputs. Figure B1-2 is a detailed depiction of an ACS, as described in Paragraph 3.2.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

The autonomous spacecraft shall be capable of successfully performing the mission function for an extended period of time without ground support at a specified level of conflict. Specifically:

- (1) The spacecraft shall operate without performance degradation for up to 60 days from the last initialization update.
- (2) The spacecraft shall operate for up to 6 months from the last initialization update, within acceptable performance degradation limits, for mission-prioritized functions as defined by each mission.

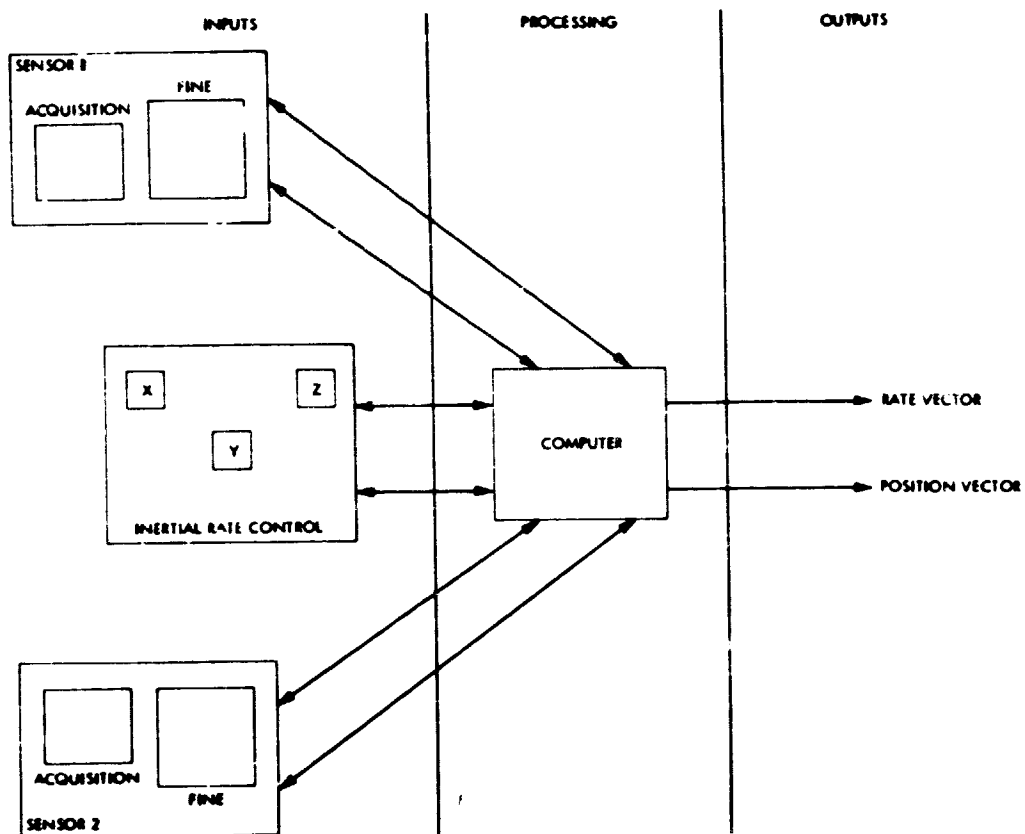


Figure B1-1. Input/Output Block Diagram

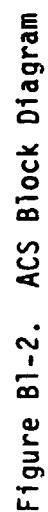


Figure B1-2. ACS Block Diagram

3.2

SPACECRAFT REQUIREMENTS

The following generalized model of a spacecraft is used as reference for the algorithm given below in Section 5. Such a model should meet the 60 day/6 month requirement specified in Paragraph 3.1.

Assume a spacecraft with 3-axis active control. In general, two celestial references must be obtained in order to maintain spacecraft control. These references may be selected from sun, earth or star. Once such a reference is acquired, fine sensors will, in general, provide two axes of control. The spacecraft is assumed to be equipped with fine sensors, which provide such two-axis control using each of two celestial references.

Two independent and functionally redundant control schemes may be developed using two distinct celestial references. If these references are denoted by A and B, then one scheme uses fine sensors for A to control two axes and a fine sensor for B to control the third axis. The other scheme uses fine sensors for B to control two axes and a fine sensor for A to control the third axis. Each such control scheme will be referred to as a 'reference scheme'. The spacecraft is assumed to have two reference schemes.

In such a reference scheme, one celestial reference can be used to control two axes. This reference will be denoted as '2XR'. The remaining celestial reference in the scheme will be denoted, by analogy, as '1XR'.

The spacecraft is assumed to be equipped with acquisition sensors, each sensing the presence of a celestial reference. A fine sensor provides rate and position information for a given axis. The spacecraft may have redundant acquisition sensors and electronics, but must be assumed equipped with redundant fine sensors and electronics for each axis in a pair controlled by a celestial reference.

The spacecraft is assumed to have a system for controlling rates about each axis, a system independent of the celestial references used in the reference schemes. This system is referred to as the 'inertial rate control' system. It must provide axial rate control sufficient to maintain the acquisition of celestial reference.

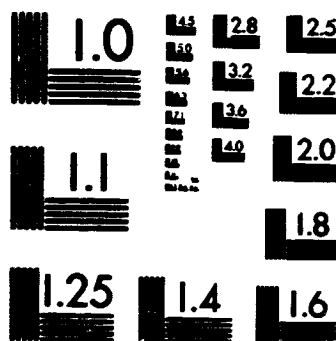
The attitude control system (ACS) of this spacecraft must have the ability to switch in and swap out redundant components, such as sensors, electronics, and redundant processor/memory units. In addition, the ACS must be able to configure to a redundant reference scheme and to selectively use inertial rate control for any axis.

The axioms and definitions given above are presented in Tables B1-1 and B1-2.

83

6920

3/B



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

Table B1-1. Axioms for a 3-Axis Stable Spacecraft

1. TWO CELESTIAL REFERENCES ARE NEEDED FOR 3-AXIS CONTROL.
2. TWO FUNCTIONALLY REDUNDANT REFERENCE SCHEMES MAY BE DERIVED FROM THE TWO CELESTIAL REFERENCES AND ARE AVAILABLE FOR CONTROL.
3. ACQUISITION SENSORS AND ELECTRONICS ARE AVAILABLE TO SENSE THE PRESENCE OF A CELESTIAL REFERENCE.
4. REDUNDANT FINE SENSORS AND ELECTRONICS ARE AVAILABLE TO CONTROL RATE AND POSITION FOR EACH AXIS.
5. INERTIAL RATE CONTROL, INDEPENDENT OF THE TWO CELESTIAL REFERENCES, IS AVAILABLE FOR EACH AXIS, SUFFICIENT TO CONTROL RATES WHILE ACQUISITION OF CELESTIAL REFERENCE IS MAINTAINED.
6. THE ACS IS EQUIPPED WITH REDUNDANT PROCESSOR/MEMORY UNITS.
7. THE ACS CONTROLS THE CONFIGURING TO REDUNDANT UNITS.

Table B1-2. Definitions

REFERENCE SCHEME:	TWO CELESTIAL REFERENCES USED TO CONTROL 3 AXES OF SPACECRAFT, ONE REFERENCE IS USED TO CONTROL TWO AXES; THE OTHER TO CONTROL REMAINING AXIS.
2XR:	A CELESTIAL REFERENCE IN A REFERENCE SCHEME USED TO CONTROL TWO AXES.
1XR:	A CELESTIAL REFERENCE IN A REFERENCE SCHEME USED TO CONTROL ONE AXIS.
CELESTIAL REFERENCE:	ONE SELECTED FROM SUN, EARTH OR STAR.
ACQUISITION SENSOR:	A SENSOR USED TO SENSE THE PRESENCE OF A CELESTIAL REFERENCE.

SECTION 4

SUBSYSTEM FUNCTIONAL OPERATIONS

The reference loss and reacquisition algorithm is presented in four parts: initial acquisition, 2XR reacquisition, 1XR reacquisition and fine sensor fault analysis. Each may be represented as a software routine initiated and monitored by an executive routine.

The executive routine would have the responsibility of scheduling and timing each of the four routines, and could set the spacecraft in a mode necessary for the initial acquisition of celestial references. When called, the initial acquisition routine will either acquire the two references or return to the executive routine, having established a failure mode of spacecraft operation.

Under the normal mode of spacecraft operation, defined as the state in which the spacecraft uses fine sensors of celestial reference for 3-axis attitude control, the executive routine would periodically call the 2XR reacquisition routine twice, followed by the 1XR reacquisition routine, as a health check for the sensors. However, a more likely reason for calling these routines would be as a response to an attitude anomaly. Typically, a control law would periodically be executed by the executive routine. This law would use rate and estimated position vectors to compute an update to the estimated position of the spacecraft. A significantly anomalous difference between old and new estimated position vectors or a rate vector in magnitude exceeding spacecraft specifications or pointing accuracy requirements would lead to an analysis of and recovery from the apparent fault in attitude control. The executive routine, which determines that this fault has occurred, would call the 2XR, then 1XR reacquisition routines as a part of its reaction to this fault.

The 2XR reacquisition routine recovers 2-axis attitude control. It reacts to problems in either acquisition or fine sensors by switching to redundant units or, failing to correct the problem this way, establishing a failure mode of spacecraft operation. The correction procedure may involve initiating a call of the initial acquisition routine, if a complete loss of reference has occurred.

A less drastic recovery procedure may involve a call to the 1XR reacquisition routine, after 2-axis control has been established.

These operations of the 2XR reacquisition routine are all designed to take precedence over other operations of the attitude control subsystem. In contrast to this, the operation of switching to redundant units incorporates a scheduling of the fine sensor routine, which operates in a mode dictated by the priorities established in the executive routine. Hence, the fine sensor routine may be executed in concert with other ACS operations.

The 1XR reacquisition routine recovers control of the third axis in a reference scheme. Its processing, though much the same as that of the 2XR reacquisition routine, is recognized as being less urgent, since 2-axis control is assumed established. Hence, this processing is carried out in an

'open loop' fashion with the executive routine acting as a monitor. The processing of the 1XR reacquisition routine includes acquisition of 1XR, fine sensor fault isolation, management of redundant components and in the worst case, establishing a failure mode of spacecraft operation. If a fine sensor has been determined to be responsible for the attitude problem, the fine sensor routine is scheduled.

The fine sensor routine compares a presumed faulty sensor with the redundant fine sensor, switched in for control. A faulty fine sensor is restricted from again being used as a fine sensor for control. However, if the fault in the fine sensor is merely a transient, then this faulty sensor is simply turned off and returned to the pool of sensors which can be used for control.

4.1 FAILURE MODES

In the above paragraph certain failure modes have been indicated as terminal states of three of the four routines. These modes represent the best chance for spacecraft attitude control in the face of persistent faults. These modes may be listed in order of seriousness as follows:

- 1 axis degrade
- 2/3 axis degrade
- 1 reference failure
- 2 reference failure

A definition for each of these modes is presented in Table B1-3. The degrade modes represent inertial rate control constrained by the requirement of maintaining the acquisition of celestial reference. The reference failure modes are the result of faults in reference acquisition. Whenever a choice has to be made, the failure mode that best maintains the acquisition of the two celestial references is established. This is a prevailing theme throughout each of the routines, which comprise the loss of reference and reacquisition algorithm.

4.2 REDUNDANCY MANAGEMENT

Three of the four routines mentioned in paragraph 4.0 perform some sort of switching to redundant units in the face of sensor faults. This switching generally takes place in the following order:

- sensors
- electronics
- reference schemes
- sensors
- electronics
- processor/memory units

Table B1-3. Definition of Failure Modes

- 1 **AXIS DEGRADE MODE:** BOTH CELESTIAL REFERENCES ARE ACQUIRED. FINE SENSORS CONTROL POSITION AND RATES ABOUT TWO SPACECRAFT AXES. THE INERTIAL RATE CONTROL IS USED TO CONTROL RATES ABOUT THE THIRD AXIS.
- 2/3 **AXIS DEGRADE MODE:** BOTH CELESTIAL REFERENCES ARE ACQUIRED BUT FINE SENSORS CONTROL POSITION AND RATE FOR AT MOST ONE AXIS. THE INERTIAL RATE CONTROL IS USED TO CONTROL RATES ABOUT THE REMAINING AXES.
- 1 **REFERENCE FAILURE MODE:** ONE CELESTIAL REFERENCE IS ACQUIRED, THE OTHER IS NOT. INERTIAL RATE CONTROL IS USED TO CONTROL RATES ABOUT ALL AXES.
- 2 **REFERENCE FAILURE MODE:** NO CELESTIAL REFERENCE IS ACQUIRED. THE SPACECRAFT IS POINTING IN SOME UNDETERMINED DIRECTION AND RATE ABOUT EACH AXIS IS CONTROLLED BY THE INERTIAL RATE CONTROL.

Once a processor swap has been ordered, memory of past switching to redundant components has been lost. Hence the above sequence would begin again until some set of sensor, electronics, reference scheme and processor/memory yields a set which corrects the problem. A number of choices of such sets is available, depending on the level of redundancy per item. (In the diagram of an ACS given in Figure B1-2, 16 choices are possible.)

In some cases a sequence of switching to redundant components is inhibited, if such a switch poses a problem in maintaining reference acquisition. This is particularly true in the case of fine sensor fault isolation logic, when a switch in circuitry may lead to correction of a fine sensor problem, but jeopardize the state of the acquisition sensors. In general, a switch from electronics numbered as two to electronics numbered as one is prevented.

4.3 FUNCTIONAL DATA FLOW DIAGRAM

The functional data flow diagram for the loss of reference and reacquisition algorithm is presented in Figure B1-3.

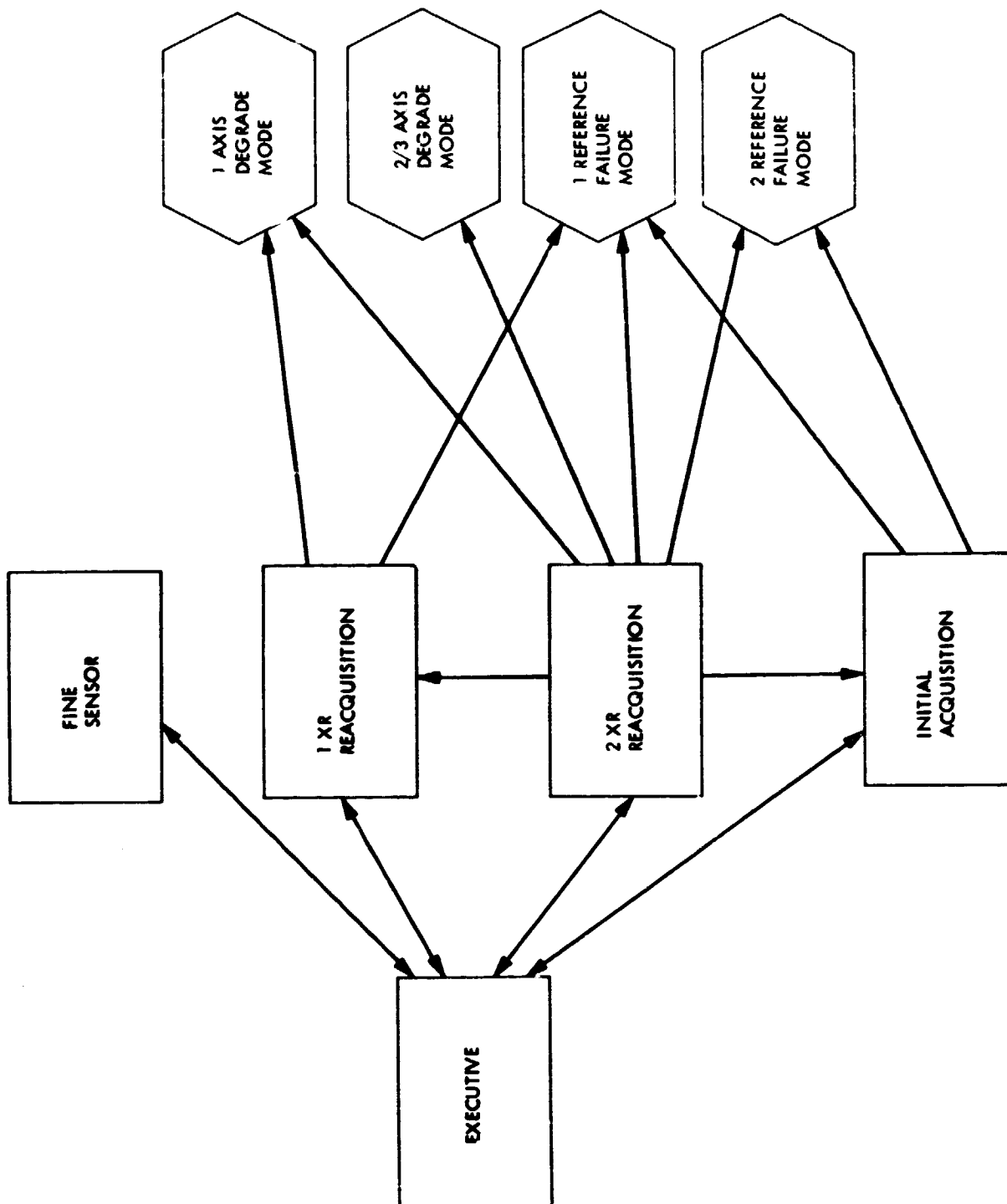


Figure B1-3. Functional Data Flow Diagram

SECTION 5

SUBSYSTEM FUNCTIONAL REQUIREMENTS

5.1 INPUTS

Each of the routines, which comprise the loss of reference and reacquisition routine, respond to sensor readings by taking certain logic paths. The executive routine calls this routine as a result of a perceived attitude anomaly, an initial sequence of operation, or a general subsystem health check.

5.2 PROCESSING

The following is a top-level description of the routines presented in Figures B1-4, B1-5, B1-6, and B1-7, which comprise the loss-of-reference and reacquisition algorithm. A precise description of the process or decision blocks has been omitted, given the constraint of the generality of the model to which these algorithms are meant to apply. In addition to the remarks given below, a narrative is presented in the figures.

5.2.1 Initial Acquisition

After inertial rate control is established for each axis, further processing of this routine is delayed until the spacecraft is ready to search for the two celestial references. One of the two reference schemes is nominally used for initial acquisition. The search for the 2XR is timed with a software timer, which is set at launch. However, future use of this routine (see 2XR reacquisition) may require an on-board computation of the optimal timing of celestial reference acquisition. The acquisition itself involves a search sequence followed by a check of the acquisition sensor. It is the non-null output of this sensor which determines the presence of the 2XR reference. (Correspondingly, the null output of this acquisition sensor defines celestial reference loss.)

The acquisition of 2XR leads to the use of fine sensors for the control of two axes. The search for 1XR may then proceed in a fashion similar to the search for 2XR, although the search sequence will be contingent upon the need to preserve 2XR acquisition. After 1XR is acquired the normal mode of spacecraft operations may be established and this routine ends with a return to the executive routine.

If 2XR is acquired during the allotted time but 1XR is not, the sensors and circuitry are checked. A chart, which contains information concerning powered state, 'in use' state, fault state and scheme/axis use for each sensor, may be kept on board to track sensor usage and provide

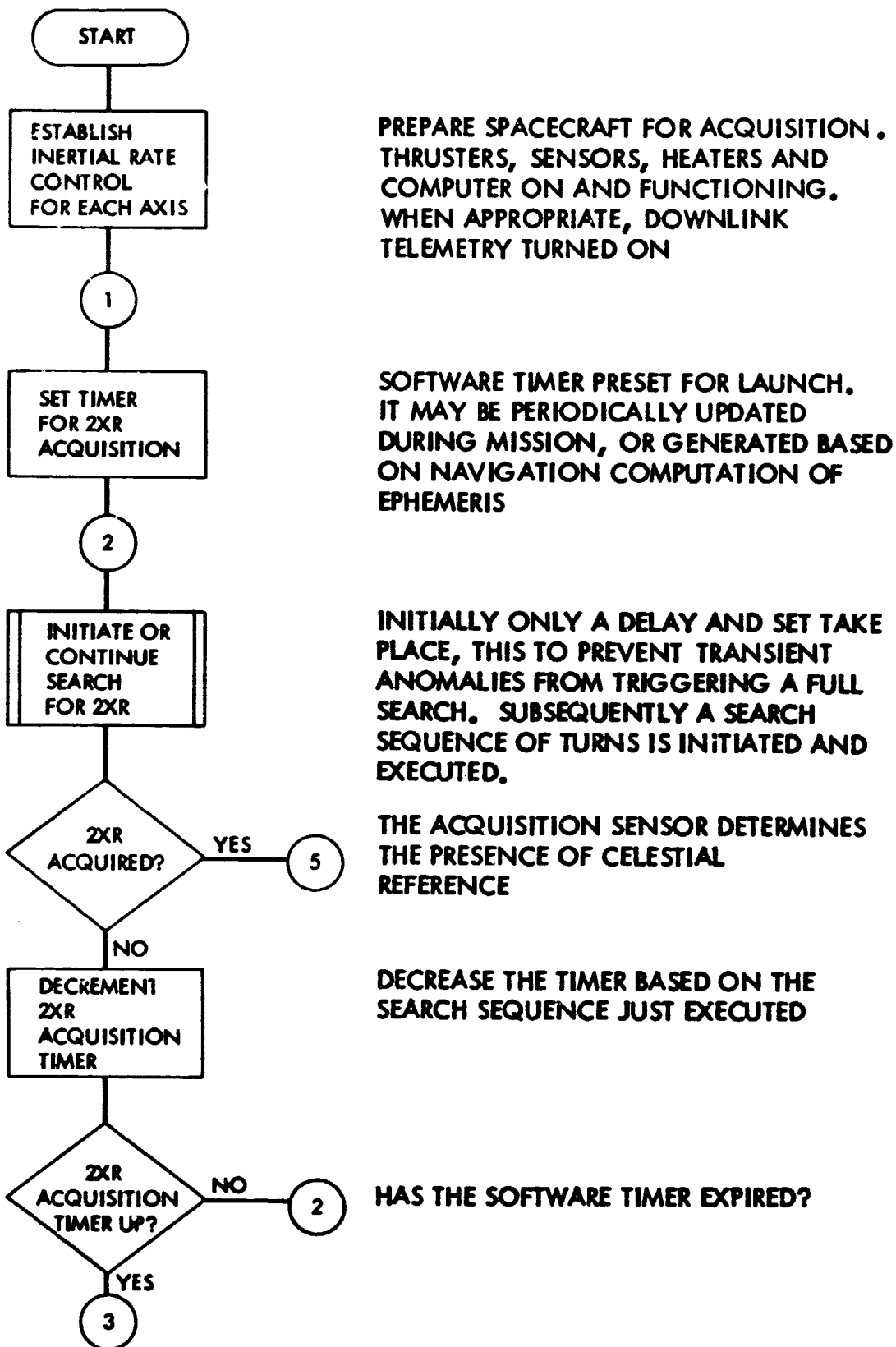


Figure B1-4. Initial Acquisition (Sheet 1 of 4)

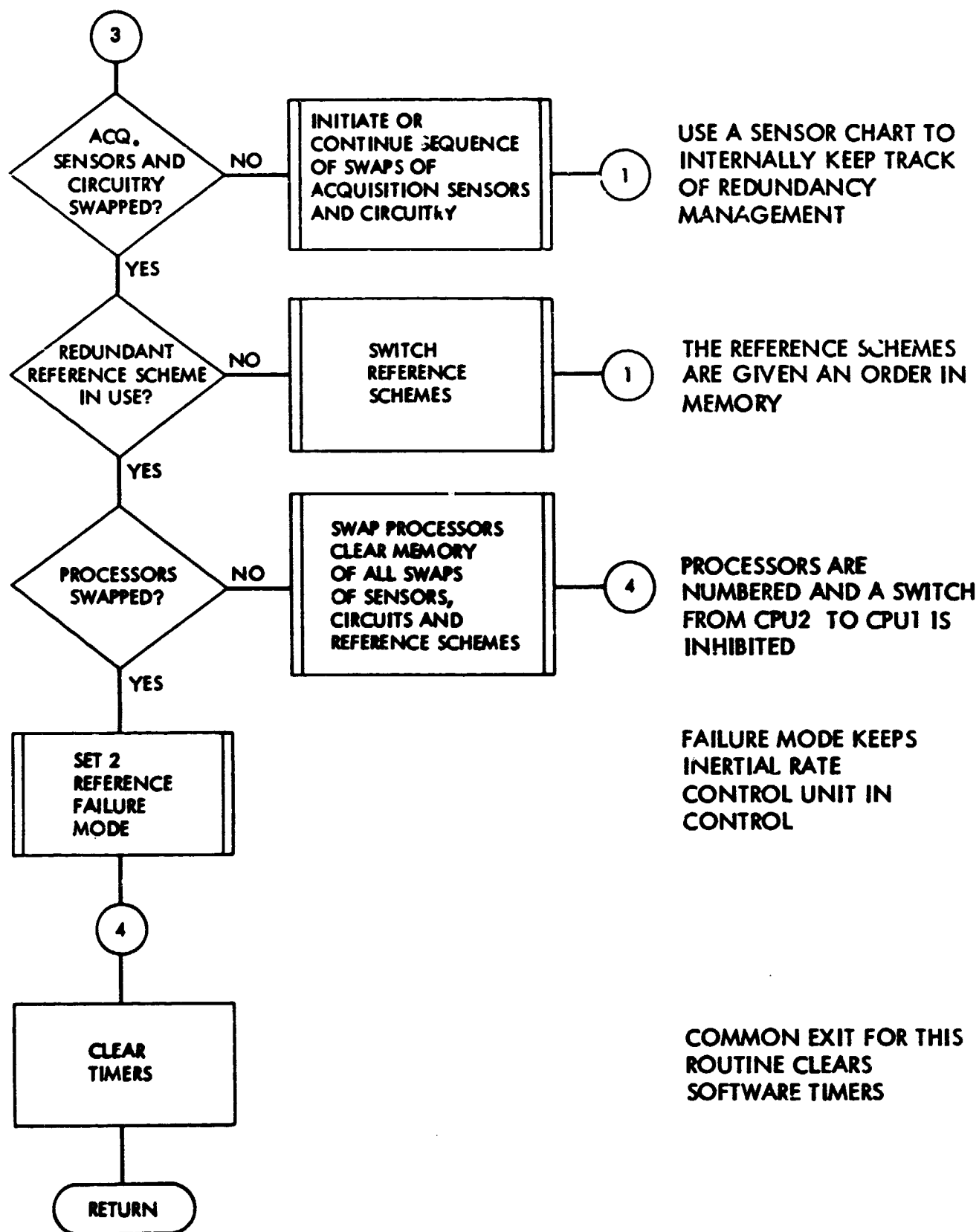
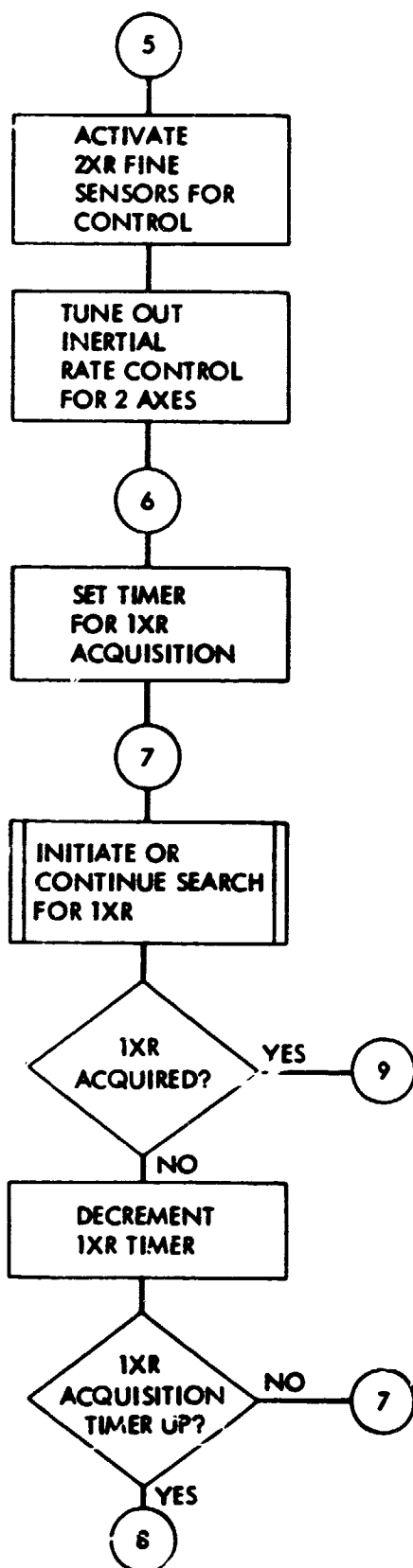


Figure B1-4. Initial Acquisition (Sheet 2 of 4)



SET AXIAL FINE SENSORS FOR CONTROL. WAIT FOR TRANSIENTS TO DIE OUT BEFORE SWITCH OFF OF INERTIAL RATE CONTROL AND SWITCHING IN FINE SENSORS FOR CONTROL

SOFTWARE TIMER SET EITHER BY A PRELAUNCH TIMING OF A SEARCH PATTERN OR BY NAVIGATION COMPUTING OF EPHEMERIS

INITIALLY WAIT, THEN PERFORM A SERIES OF PREPLANNED TURNS

READ THE 1XR ACQUISITION SENSOR TO DETERMINE THE PRESENCE OF CELESTIAL REFERENCE

DECREASE THE TIMER BASED ON THE SEARCH SEQUENCE JUST EXECUTED

HAS THE SOFTWARE TIMER EXPIRED?

Figure B1-4. Initial Acquisition (Sheet 3 of 4)

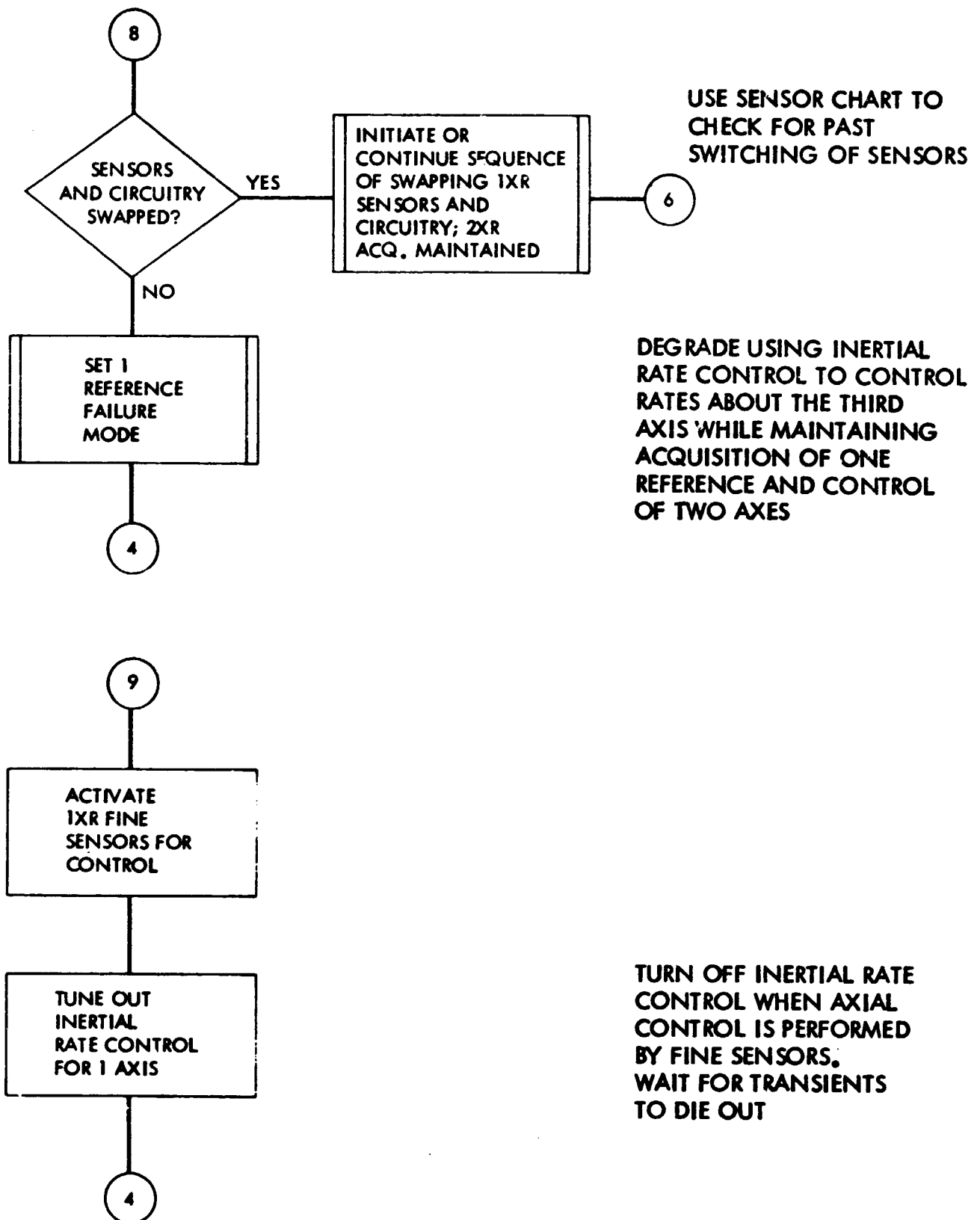
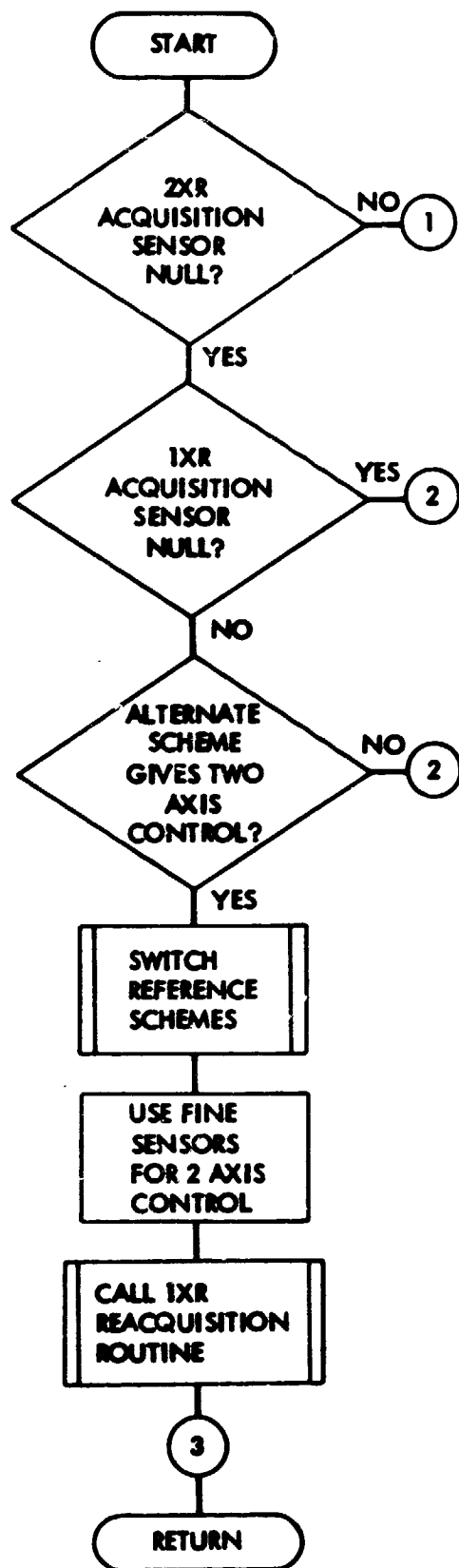


Figure B1-4. Initial Acquisition (Sheet 4 of 4)

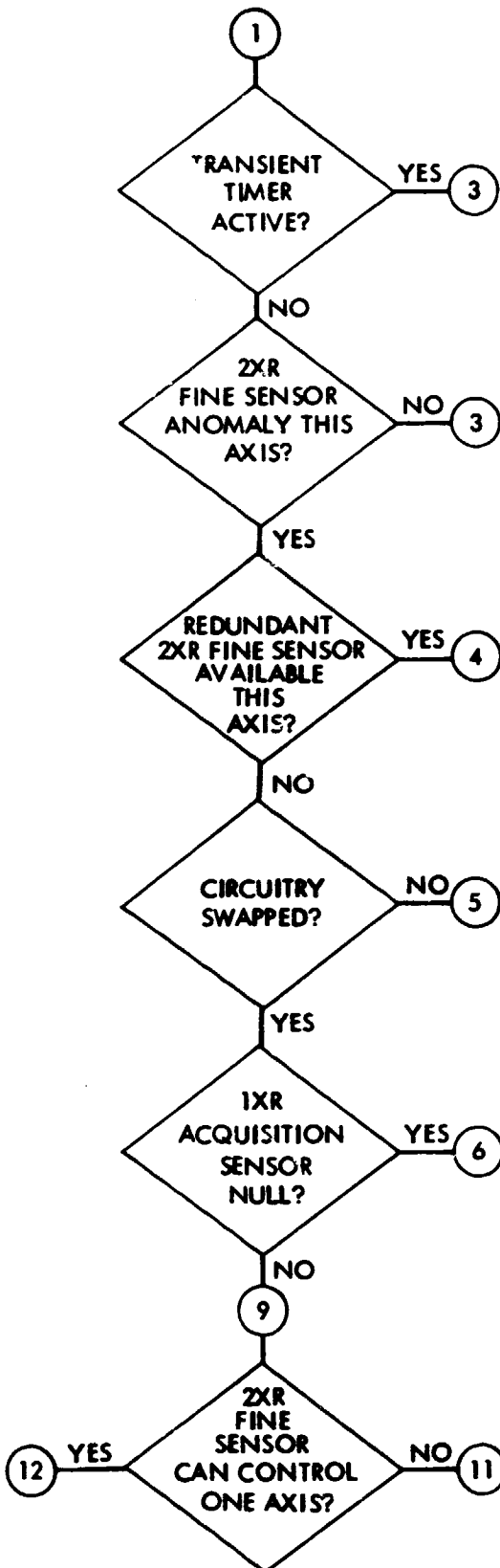


THE ACQUISITION SENSOR DETERMINES THE LOSS OF REFERENCE. CHECK FOR POSSIBLE 'GLITCHES' IN THE SENSOR

USE SENSOR CHART TO DETERMINE IF AN ALTERNATE SCHEME IS AVAILABLE FOR THIS CONTROL

START THE ACQUISITION OF 1XR

Figure B1-5. 2XR Reacquisition Routine (Sheet 1 of 6)



USE A SOFTWARE TIMER TO TIME THE TRANSITION FROM CONTROL OF ONE FINE SENSOR TO A REDUNDANT SENSOR. A CALIBRATION MAY BE NEEDED DURING THIS TIME.

READ FINE SENSOR. CHECK OUTPUT SIGNAL WITHIN ACCEPTABLE LIMITS. IF A PROBLEM, WAIT FOR TRANSIENT 'GLITCH' TO CLEAR.

CHECK SENSOR CHART FOR POSSIBLE REDUNDANT SENSOR.

CIRCUITRY INTERNALLY NUMBERED. SWITCH FROM ELECTRONICS 2 TO ELECTRONICS 1 IS INHIBITED.

WITH ACQUISITION SENSOR READING 'REFERENCE LOSS', TWO AXES NOT CONTROLLED.

AT THIS POINT 2XR FINE SENSORS CAN CONTROL AT MOST ONE AXIS. CHECK SENSOR CHART FOR POSSIBLE SWITCH IN REFERENCE SCHEMES.

Figure B1-5. 2XR Reacquisition Routine (Sheet 2 of 6)

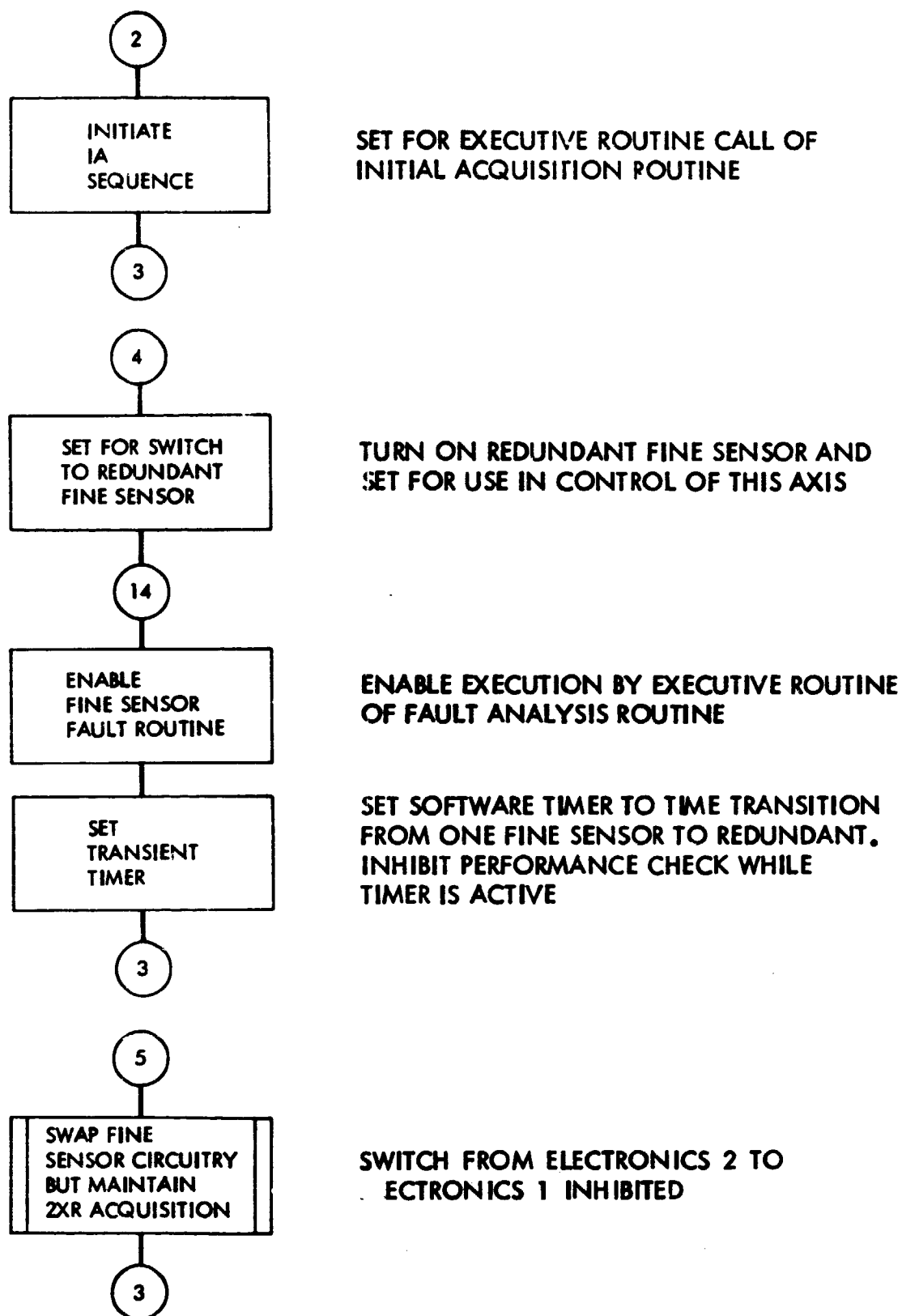


Figure B1-5. 2XR Reacquisition Routine (Sheet 3 of 6)

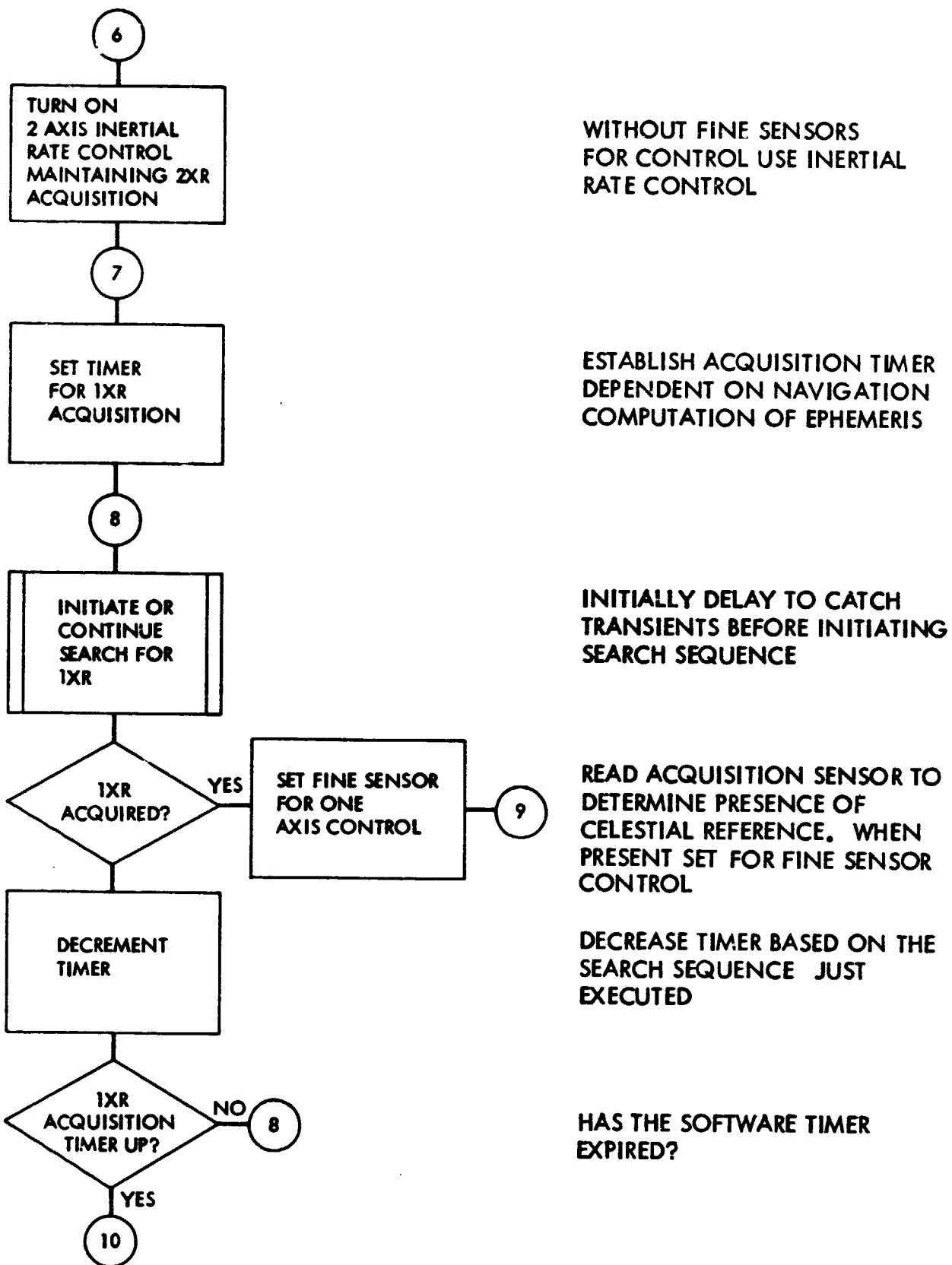


Figure B1-5. 2XR Reacquisition Routine (Sheet 4 of 6)

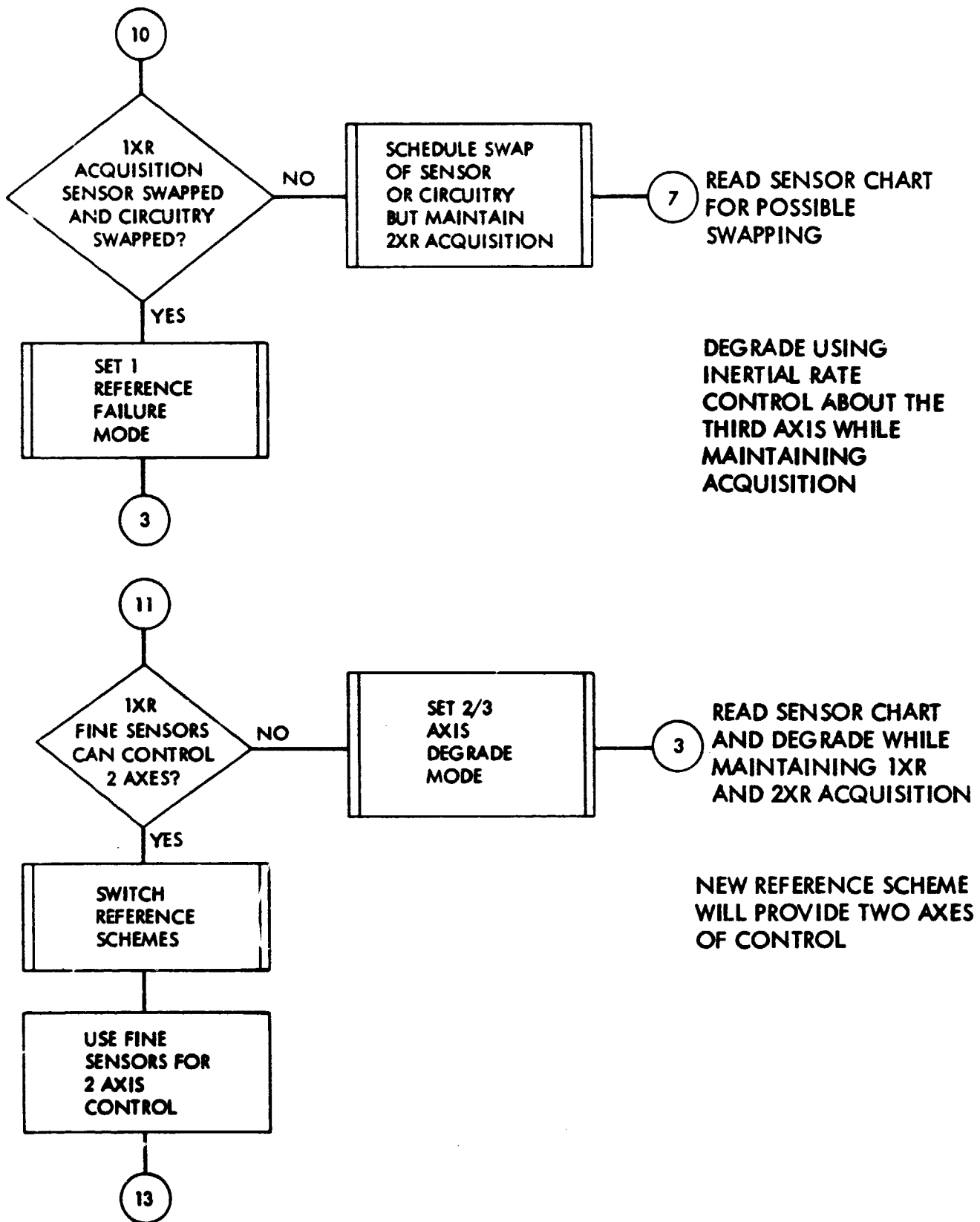


Figure B1-5. 2XR Reacquisition Routine (Sheet 5 of 6)

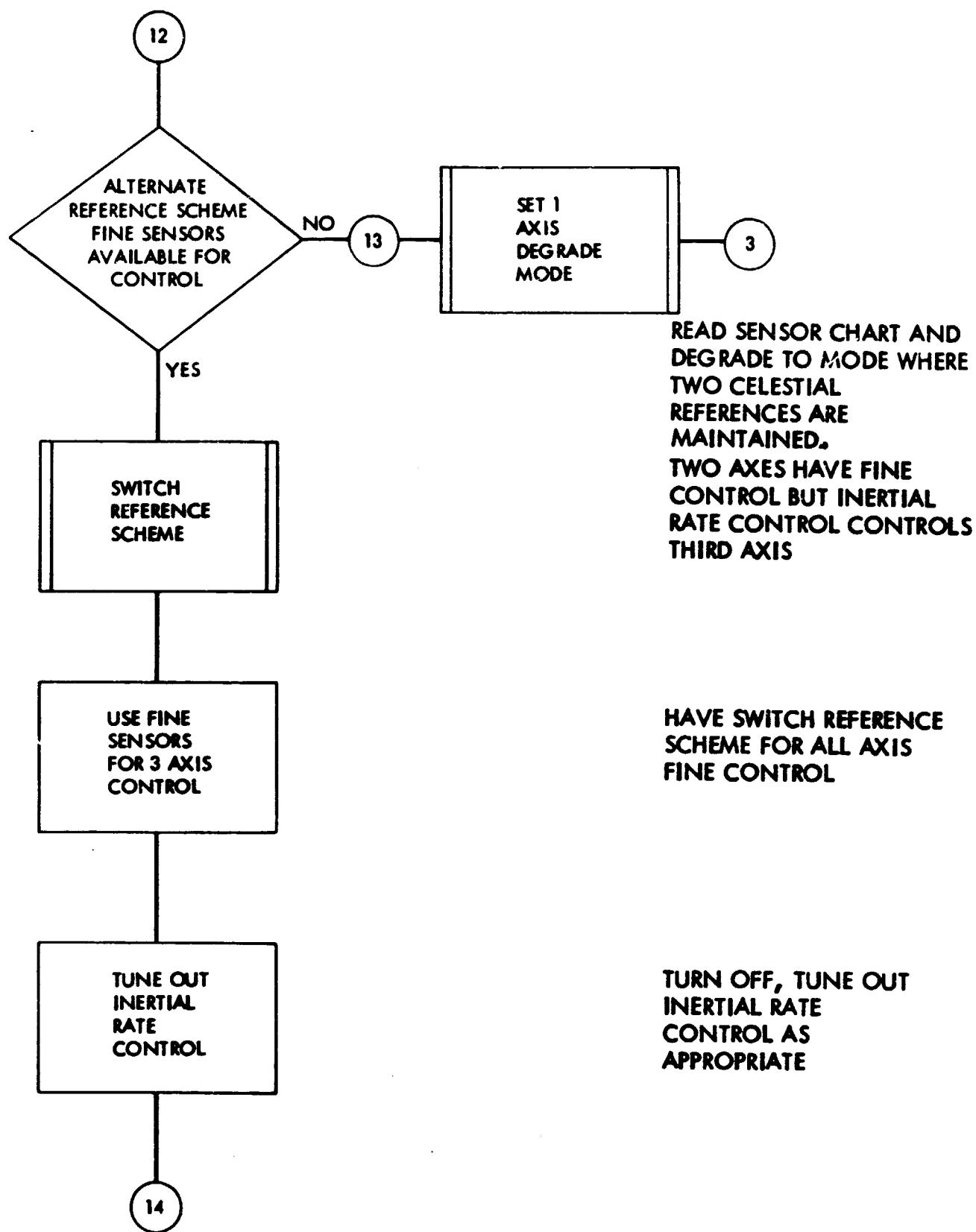


Figure B1-5. 2XR Reacquisition Routine (Sheet 6 of 6)

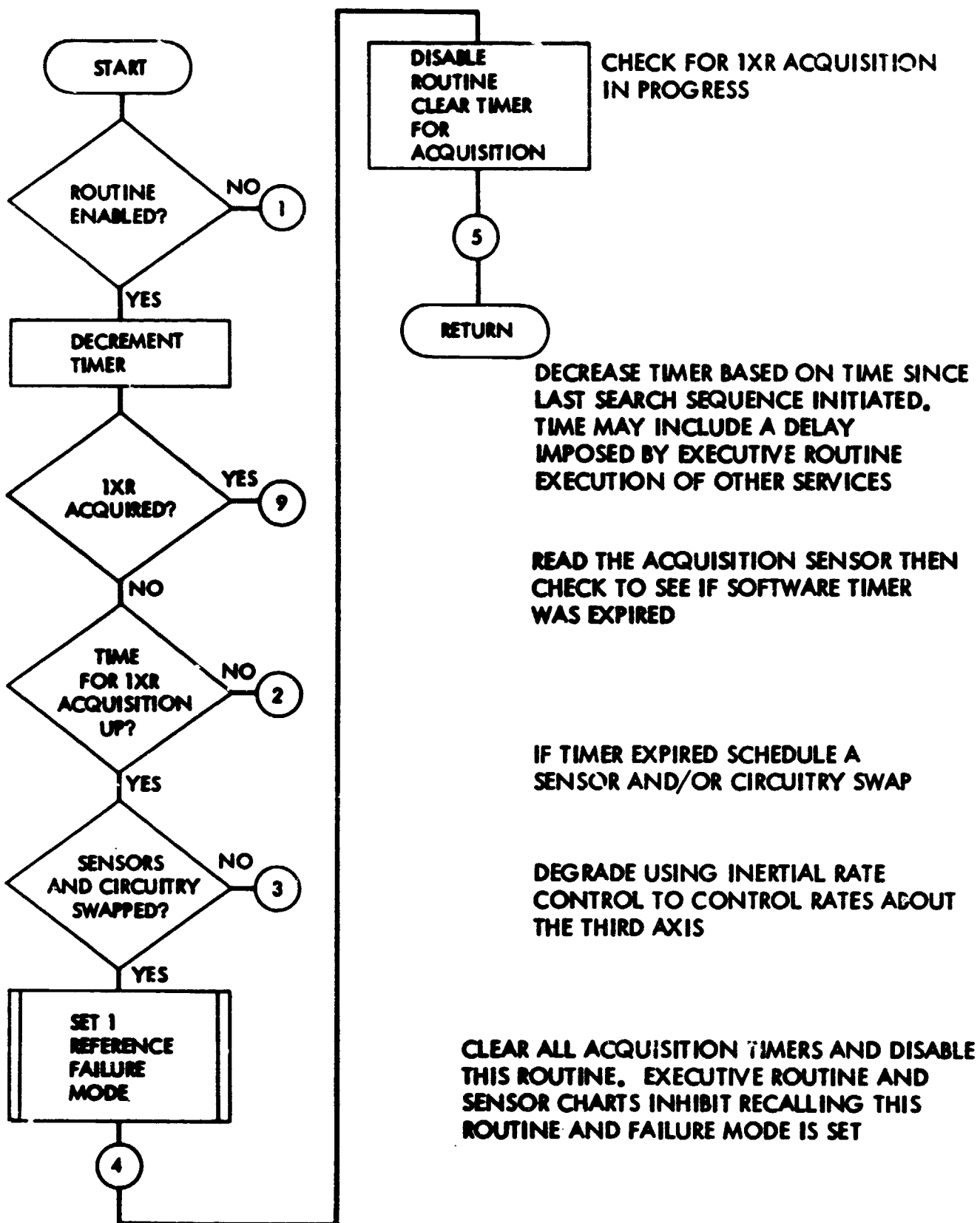
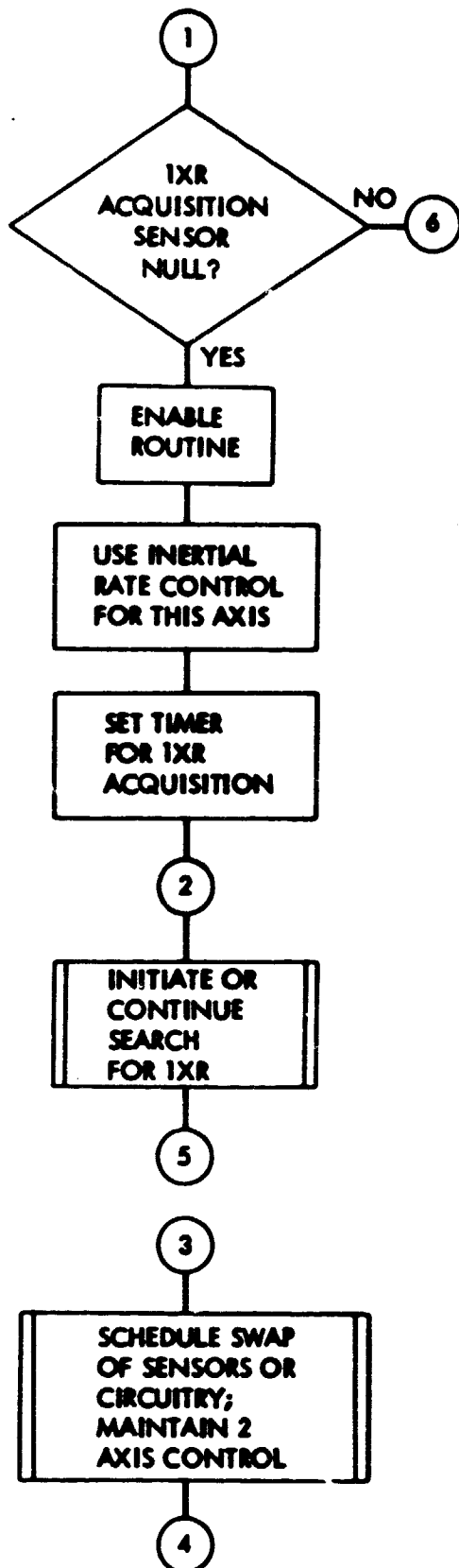


Figure B1-6. 1XR Reacquisition Routine (Sheet 1 of 4)



READ STATE OF ACQUISITION SENSORS TO DETERMINE PRESENCE OF REFERENCE. CHECK FOR POSSIBLE 'GLITCHES' IN SENSOR. SET FLAG FOR OPEN LOOP SEARCH FOR 1XR BY THIS ROUTINE

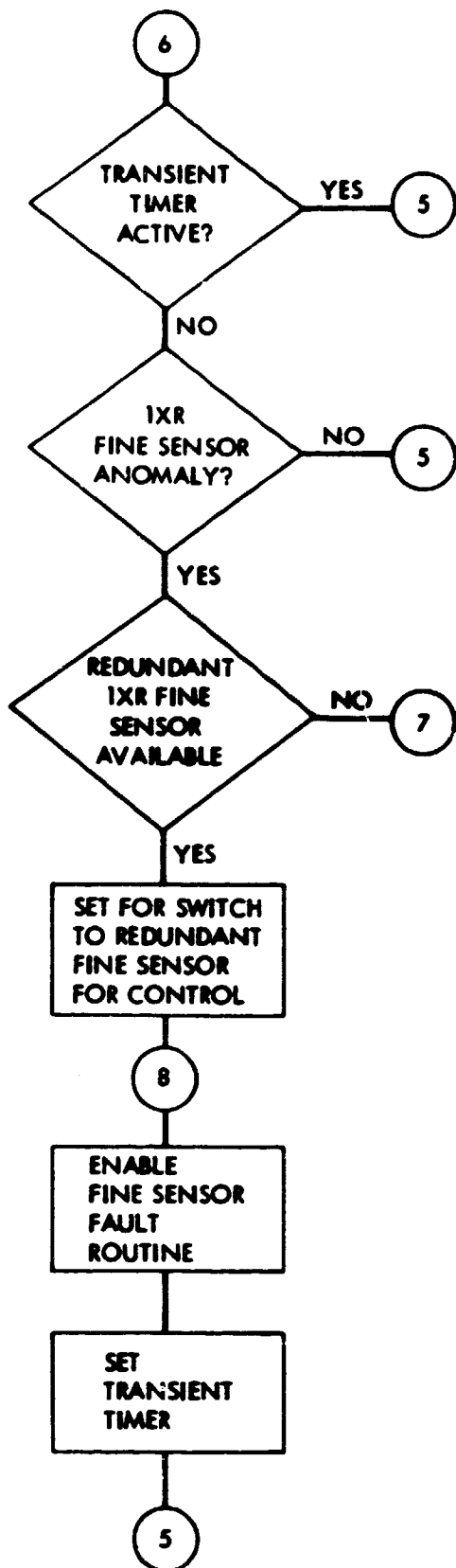
USE INERTIAL RATE CONTROL TO CONTROL RATES ABOUT THIS AXIS

SET SOFTWARE TIMER TO TIME THE ACQUISITION SEQUENCE

INITIALLY DELAY TO PROTECT FURTHER FROM TRANSIENTS. SUBSEQUENTLY, PERFORM A SEARCH SEQUENCE FOR CELESTIAL REFERENCE

CIRCUITRY SWAPPED ONLY WHILE ACQUISITION OF 2XR MAINTAINED

Figure B1-6. 1XR Reacquisition Routine (Sheet 2 of 4)



INHIBIT PERFORMANCE CHECK OF FINE SENSOR IF A REDUNDANT SENSOR SWITCHED INTO USE

READ FINE SENSOR. CHECK OUTPUT SIGNAL WITHIN ACCEPTABLE LIMITS. IF A PROBLEM WAIT FOR TRANSIENT 'GLITCH' TO CLEAR BEFORE CONTINUING

CHECK SENSOR CHART FOR REDUNDANT UNIT

WHEN AVAILABLE SET FOR SWITCH TO REDUNDANT UNIT IN CONTROL OF THIS AXIS

ENABLE EXECUTION BY EXECUTIVE ROUTINE OF FAULT ANALYSIS ROUTINE

SET SOFTWARE TIMER TO TIME TRANSITION FROM ONE FINE SENSOR TO REDUNDANT

Figure B1-6. 1XR Reacquisition Routine (Sheet 3 of 4)

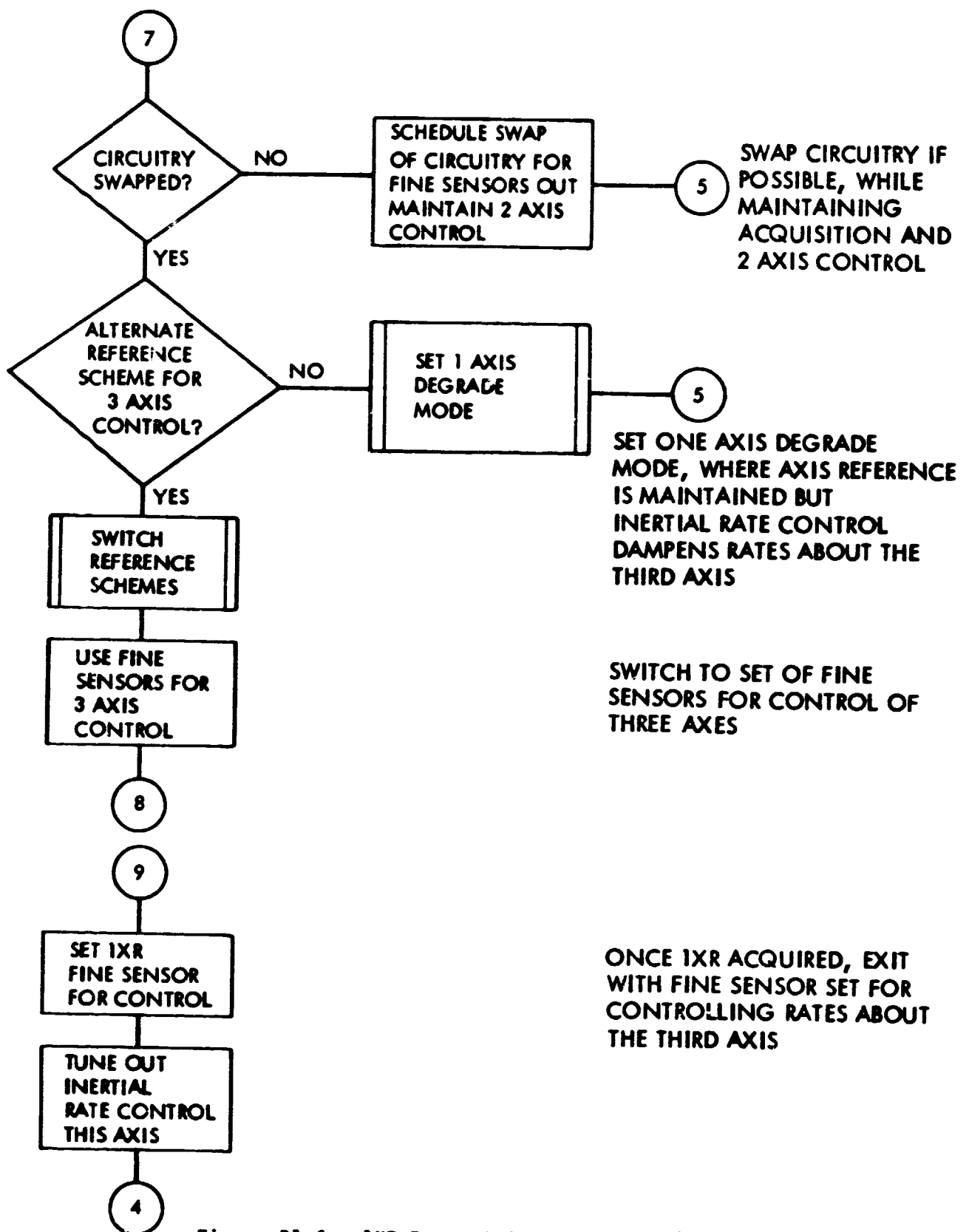


Figure B1-5. 1XR Reacquisition Routine (Sheet 4 of 4)

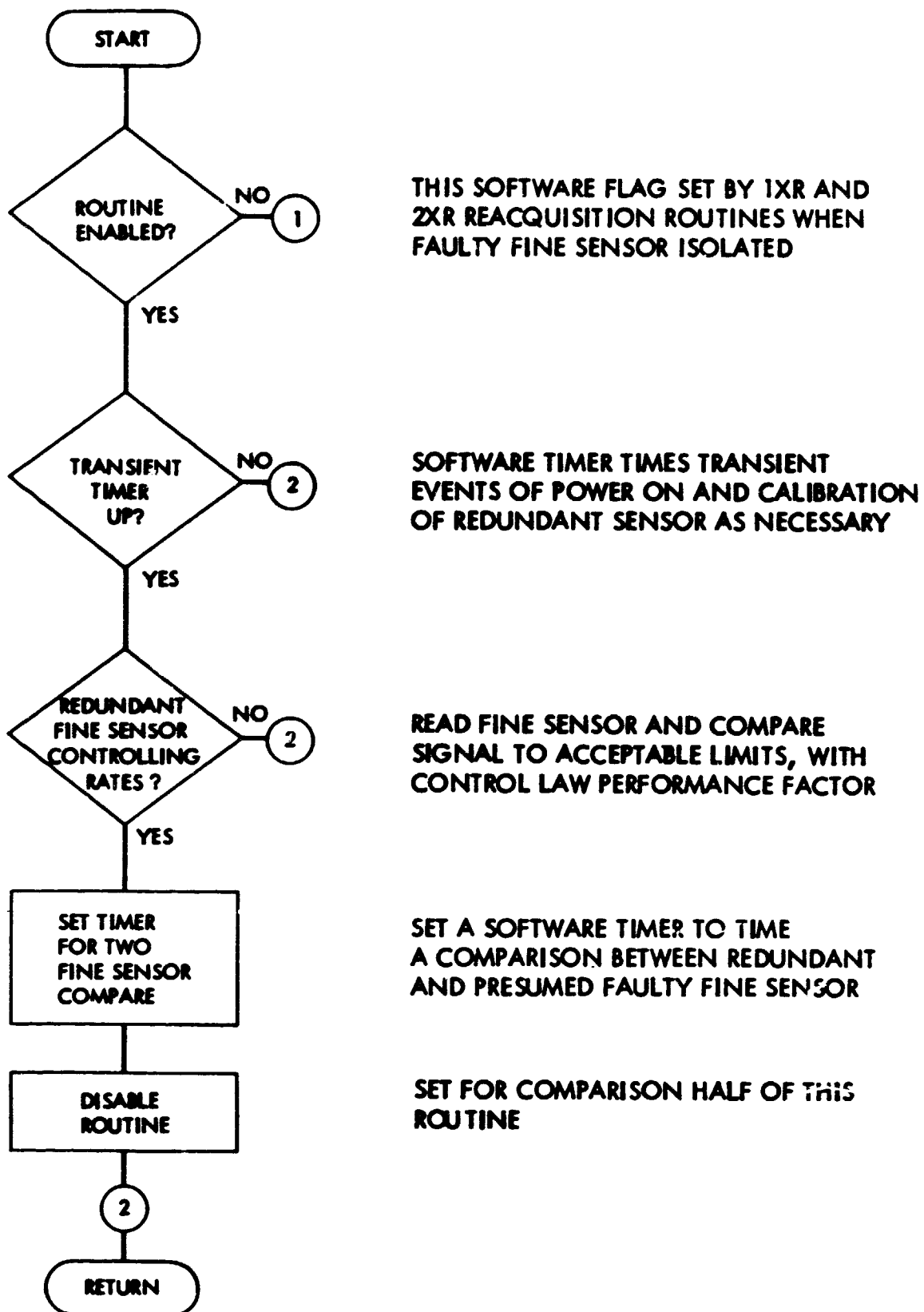


Figure B1-7. Fine Sensor Routine (Sheet 1 of 2)

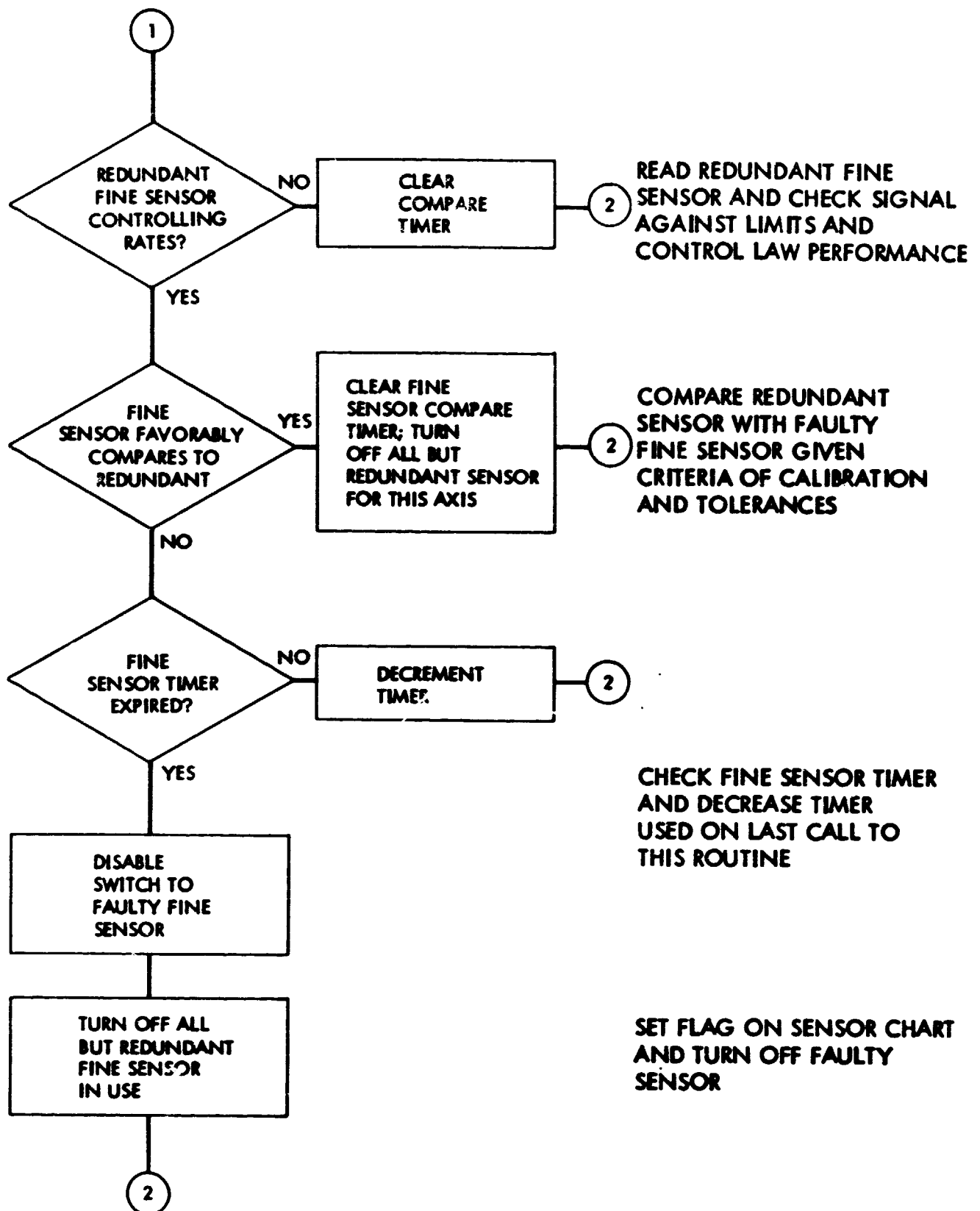


Figure B1-7. Fine Sensor Routine (Sheet 2 of 2)

information for decisions such as sensor swapping. A switch to redundant circuitry may be inhibited here, if a previous circuitry swap was made. If all sensor and circuitry swaps have been exhausted, the one-reference failure mode is established. This mode leaves the inertial rate control in control of rates about the third axis.

If the 2XR acquisition sequence times out, a more elaborate procedure of swapping and reconfiguring to redundant units ensues (see the list and remarks in paragraph 4.2). Only after a processor/memory swap has failed to clear the acquisition problem is the two reference failure mode established. This mode leaves the inertial rate control in control of rates about all axes.

5.2.2 2XR Reacquisition

The state of the acquisition sensors defines reference loss. If the 2XR acquisition sensor is null, a check for a possible sensor transient anomaly is made. If the sensor still reads null, a check of the state of the 1XR acquisition sensor determines if an immediate change of reference scheme will return at least two axes of control. If the sensor chart indicates that such control cannot be established or if both references are lost, the initial acquisition of references is reinitiated. The executive routine will set for this interruption of other services.

If a switch in reference schemes will lead to immediate two-axis control and the potential for three-axis control, then such a logic path is taken. It should be noted that the 1XR reacquisition routine, which is called a part of this 'change of reference' path, contains a potential switch in reference schemes. Hence, if only one reference scheme provides normal mode 3 axis control, this scheme will eventually be re-established after 2XR is lost.

If the 2XR acquisition sensor indicates that this celestial reference is maintained, then the fine sensors for a given axis of control are checked. A problem with such a sensor is not presumed serious enough for an immediate switch to a redundant unit. A delay in further fault correction procedures is taken in order to allow for both a transient 'glitch' and a power on and calibration of the redundant sensor. In addition, while the transient timer (a software timer which causes the aforementioned delays) is active, certain actions may be taken to reestablish fine sensor control. For example, if the fine sensor were a star tracker, the transient timer may time a flyback and sweep sequence.

Even if 1XR is acquired, the potential exists for establishing a 1 or 2/3 axis degrade mode, if all three axes cannot be controlled with fine sensors. The least serious failure mode will be established in this event (see paragraph 4.1).

If a combination of sensor, circuitry and reference schemes can be found to provide three-axis control, such a combination will be found and the normal mode of spacecraft operation re-established. Whenever a redundant sensor is scheduled for use, the fine sensor fault analysis routine is enabled and the transient timer is set before a return to the executive routine.

5.2.3 1XR Reacquisition

The 1XR reacquisition routine may be called either by the 2XR reacquisition routine or by the executive routine. The main branches of logic in this routine are divided by the state of the 1XR acquisition sensor. The loss of 1XR as determined by this sensor initiates a search sequence once a delay is executed to check for possible sensor 'glitches'.

Since two-axis control is established, the search for 1XR may be made in an open loop fashion. The executive routine may perform other services and may terminate this sequence, if two-axis control is in jeopardy. This open loop processing is also apparent in the manner in which sensors and circuitry are swapped in case 1XR is not acquired by the search sequence. The usual structure of preservation of 2XR acquisition applies in any such swapping sequence with the one-reference failure mode a possible terminal mode, if such swapping fails to help recover 1XR.

If 1XR is acquired, normal three-axis control is re-established. Subsequent use of this routine in the presence of 1XR checks the performance of the fine sensors for controlling the third axis. The processing here closely resembles similar processing in the 2XR reacquisition routine. In the event of a fault with the fine sensor, a combination of sensor, circuitry and reference scheme providing three-axis normal control will be found. Otherwise the one-axis degrade mode will be established as a failure mode in case of a persistent fault. The open loop processing mentioned previously again becomes apparent during much of the fault correction processing.

After a faulty sensor has been isolated and a redundant sensor scheduled to replace it, the fine sensor fault analysis routine is scheduled to test the sensor. The fine sensor check of the redundant sensor in the 1XR reacquisition routine is inhibited until certain transients have died down as timed by the transient timer. This routine returns to the executive routine when the fine sensor is determined to be working properly.

5.2.4 Fine Sensor Fault Analysis

This routine is enabled by the 2XR and 1XR reacquisition routines in the presence of a faulty fine sensor. The processing is inhibited until the transient timer is exhausted. By that time the redundant fine sensor brought into use should be controlling rates about the given axis and may properly be used to compare against the performance of the presumed faulty fine sensor.

A new timer is set for this comparison. If the faulty sensor compares favorably to the redundant sensor, it is perceived to be healthy and the problem seen as only transient. The sensor is turned off and made available for future use.

If the comparison timer expires before the faulty sensor has responded, this sensor is turned off and flagged so as not to be used again for control. Since the processing in this routine is open loop, the executive routine may be performing other services, including further fault isolation, if the redundant fine sensor fails to perform properly. Eventually, a fine sensor is found to control axis rates and when this happens all faulty sensors will be switched off. Of course, in the event of a failure mode, the fine sensors which were used for controlling the rates about a given axis will also be switched off.

The flags set on the sensor chart are software flags, and as such are processor specific. A swap of processors will clear the memory of such flags, thereby allowing all fine sensors another chance at achieving control. Under such circumstances, this is likely the proper course of action.

5.3 OUTPUTS

The outputs of the routines which comprise the reference loss and reacquisition algorithm consist of rate and position information used for the control law. As part of the internal processing and redundancy management of these routines, certain flags are set and certain failure modes are established.

SECTION 6

INTERFACE LIST/MATRIX

Table B1-4 lists the external interfaces of the routines comprising the reference loss and reacquisition algorithm.

SECTION 7

PERFORMANCE REQUIREMENTS

The initial acquisition routine should be scheduled for execution shortly after launch. The 2XR and 1XR reacquisition routines should be periodically executed once the normal mode of spacecraft operation is established.

SECTION 8

IMPLEMENTATION CONSIDERATIONS

8.1 SOFTWARE/HARDWARE

The routines which comprise the reference loss and reacquisition algorithm should reside in write protected memory of each processor/memory unit.

8.2 FINE SENSORS

The fine sensors for a given celestial reference may be configured in a package much like the (DSCS III) earth sensor. If so, then much of the fine sensor fault analysis logic in the 2XR and 1XR routines may be distributed to the sensor subsystem. The fault analysis in that case may be driven by a computer located in that subsystem.

The fine sensor analysis in the reference loss and reacquisition algorithm may, with this distributed processing, be confined to monitoring the power on and off state of the sensors and to performing comparisons solely in the case of two different sensor types controlling the same axis. An example of this may be the case of DSCS III comparing the performance of a presumed faulty pitch signal from the earth sensor against a similar signal from a sun sensor.

Table B1-4. Interface List (Sheet 1 of 2)

FROM/TO	WHAT	HOW	DOCUMENTATION
SENSORS/INITIAL ACQUISITION	ACQUISITION SENSOR SIGNALS	SENSOR ELECTRONICS AND COMPUTER I/O INTERFACE	FIGURE 2.1.1, PARAGRAPHS 4.0, 5.1, 5.2.1
SENSORS/2XR REACQUISITION	ACQUISITION AND FINE SENSOR SIGNALS	SENSOR ELECTRONICS AND COMPUTER I/O INTERFACE	FIGURE 2.1.1 PARAGRAPHS 4.0, 5.1, 5.2.2
SENSORS/1XR REACQUISITION	ACQUISITION AND FINE SENSOR SIGNALS	SENSOR ELECTRONICS AND COMPUTER I/O INTERFACE	FIGURE 2.1.1 PARAGRAPHS 4.0, 5.1, 5.2.3
SENSORS/FINE SENSOR FAULT ANALYSIS	FINE SENSOR SIGNALS	SENSOR ELECTRONICS AND COMPUTER I/O INTERFACE	FIGURE 2.1.1 PARAGRAPHS 4.0, 5.1, 5.2.4
EXECUTIVE/2XR REACQUISITION	TRANSIENT TIMER: A SOFTWARE TIMER	HARDWARE CLOCK	PARAGRAPH 5.2.2
EXECUTIVE/1XR REACQUISITION	TRANSIENT TIMER, ACQUISITION TIMER: SOFTWARE TIMERS	HARDWARE CLOCK	PARAGRAPH 5.2.3
EXECUTIVE/FINE SENSOR FAULT ANALYSIS	TRANSIENT TIMER, FINE SENSOR COMPARE TIMER: SOFTWARE TIMERS	HARDWARE CLOCK	PARAGRAPH 5.2.4

Table B1-4. Interface List (Sheet 2 of 2)

FROM/TO	WHAT	HOW	DOCUMENTATION
INITIAL ACQUISITION/ INERTIAL RATE CONTROL	TURN ON, TUNE OUT, TURN OFF SIGNALS	COMPUTER I/O INTERFACE AND ELECTRONICS	FIGURE 2.1.1 PARAGRAPH 5.2.1
2XR REACQUISITION/ INERTIAL RATE CONTROL	TURN ON, TUNE OUT, TURN OFF SIGNALS	COMPUTER I/O INTERFACE AND ELECTRONICS	FIGURE 2.1.1 PARAGRAPH 5.2.2
1XR REACQUISITION/ INERTIAL RATE CONTROL	TURN ON, TUNE OUT, TURN OFF SIGNALS	COMPUTER I/O INTERFACE AND ELECTRONICS	FIGURE 2.1.1 PARAGRAPH 5.2.3
INITIAL ACQUISITION/ SENSORS	SWITCH ACQUISITION SENSORS AND/OR ELECTRONICS	COMPUTER I/O INTERFACE AND ELECTRONICS	FIGURE 2.1.1 PARAGRAPHS 4.2, 5.2.1
2XR REACQUISITION/ SENSORS	SWITCH ACQUISITION OR FINE SENSORS AND/OR ELECTRONICS	COMPUTER I/O INTERFACE AND ELECTRONICS	FIGURE 2.1.1 PARAGRAPHS 4.2, 5.2.2
1XR REACQUISITION/ SENSORS	SWITCH ACQUISITION OR FINE SENSORS AND/OR ELECTRONICS	COMPUTER I/O INTERFACE AND ELECTRONICS	FIGURE 2.1.1 PARAGRAPHS 4.2, 5.2.3

8.3 NAVIGATION

An on-board auto-navigation package may allow optimizing the acquisition timers used in 2AR and 1XR reacquisition routines. The search sequences then employed would reflect the additional knowledge of spacecraft position available with such a package. Otherwise the timer and search sequence would have to be set before launch, much as the case of the initial acquisition sequence.

8.4 INERTIAL RATE CONTROL

An inertial rate control package may be a package of gyros which control rates. Three gyros of the DSCS III type could suffice for the purposes of the spacecraft assumptions given in paragraph 3.2. A more sophisticated package such as the Dry Inertial Reference Unit (DRIRU) in Voyager would of course suffice for the spacecraft assumption and may also be used for calibrating fine sensors, as suggested during the transition from one fine sensor used for control to another.

8.5 GROUND SUPPORT

If an auto-navigation package is not provided, ground support may be necessary to efficiently carry out a search for celestial references. In addition, ground support would need to provide lunar and solar eclipse predictions. If an auto-navigation package with appropriate knowledge of lunar, solar and earth ephemerides is provided, the ground need only provide support for recovery from failures. Such support may include commanded restart from failure modes and ground based fault analysis through telemetry.

The ground may also need to supply long-term spacecraft unit test and analysis support. If the mission of the spacecraft extends beyond the 6 months required as stated in Paragraph 3.1, a performance check and analysis of the sensors, circuitry and computer would need to be made in order to predict and plan for expected 'old-age' anomalies. In addition, a combination of spacecraft modeling and flight experience would be needed to time acquisition sequences throughout mission life.

SECTION 9

VALIDATION REQUIREMENTS

Adequate testing of the fault analysis systems must be provided before launch of a spacecraft equipped as required (see Paragraph 3.2) and with the reference loss and reacquisition algorithm as presented above. Such testing must include: acquisition sequences, fine sensor fault scenarios, and failure mode analysis. This type of testing should also be done to the extent possible during the mission.

A ground-based model of the spacecraft should be provided to adequately check details of ground fault analysis, commanded search sequencing and, of course, any modifications to the onboard software. Such a model should be available throughout the implementation stage before launch and used to troubleshoot problems which may arise during this period. The model should contain the basic sensor and computer hardware on-board the spacecraft and be computer-driven to the extent possible to realistically simulate all mission phases.

SECTION 10

MISSION MODELS

The following spacecraft are provided as models illustrating the ideas presented above.

10.1 VOYAGER

The reference loss and reacquisition algorithm was modeled on and generalized from the Voyager spacecraft and the programs of attitude control on-board in the Voyager Attitude and Articulation Control Subsystem (AACS). For Voyager, the celestial references are the Sun and Canopus. A reference scheme on Voyager used the Sun as 2XR and the sun sensors for control of pitch and yaw axes. The DRIRU provided inertial rate control and calibration of sensors as discussed in paragraph 8.4 above.

If Voyager had been equipped with a star tracker which would provide two axes of control, the analogy would be complete. In such an event, the other reference scheme could incorporate 2XR as Canopus, using the star tracker to control roll and yaw (with the star tracker perhaps mounted along the pitch axis) and 1XR as the Sun using the sun sensors to control pitch. Continuing with the analogy, the one-reference failure mode is the roll inertial mode. The all-axis-inertial mode represents the two-reference failure mode. The 1-axis degrade mode would be a mode commanded from the ground, to maintain Sun or Canopus pointing in the event of a failure in either the star trackers or one of the sun sensor assemblies. It is unlikely that the 2/3 axis degrade mode could be anything but the all-axis-inertial mode, given the spacecraft configuration.

As mentioned earlier, the transient timer represents the time necessary to perform flyback and sweeps with the star-tracker. The 2XR search pattern is the 4 steradian search for the Sun. The 1XR acquisition sequence involves using the roll inertial mode to do a roll search for Canopus, once the Sun is acquired.

The 1XR and 2XR reacquisition routines were patterned on the celestial sensor logic and Canopus star tracker (CST) logic routines on board in the Voyager AACS.

Figure B1-8 should be compared to Figures B1-1 and B1-2 as a point of contrast and reference of the Voyager system versus that of the generalized spacecraft model described in Paragraph 3.2.

10.2 DSCS III

The DSCS III spacecraft is equipped with two reference schemes. The Sun and the Earth represent the two celestial references. One scheme uses the Earth as 2XR with the earth sensor controlling the pitch and roll axes while the Sun is 1XR and the roll sun sensors are used to control the yaw axis. Another scheme uses the Sun as 2XR with the pitch and roll sun sensors controlling pitch and yaw axes while the Earth is 1XR and the earth sensor controls the roll axis.

Beyond this point the analogy ends, since DSCS III is not equipped with an inertial rate control system as described in Paragraph 3.2. Tip-off rates are currently controlled by the sun sensors with wide deadbands for thruster firing. Ground control supports all acquisition phases. In addition, the suggested redundancy management capability is missing in the computer on-board DSCS III. Ground control performs all sensor fault analysis as well as redundancy management.

10.3 EARTH ORBITER

An earth orbiting spacecraft, which uses the Sun as a power source, provides a different point of view as regards the hierarchy of failure modes presented in Section 4.1. If the Sun is used as one of the celestial references, preserving this reference may mean establishing a more serious failure mode, which maintains Sun acquisition, in preference to a less serious mode. Even a two-reference failure mode may be a preferred failure mode of operation in the presence of both acquisition and fine Sun sensor failures. In such a circumstance, the assumption of Sun power means that a crude sense of Sun presence may be derived from the power subsystem alone via the power output by that subsystem. Such a system may provide virtual two-axis control, if properly monitored, by using wide deadbands for thruster firing to maintain attitude around a power spike.

Of course, if the spacecraft is equipped with an Radioisotope Thermal Generator (RTG) system such a method of control with the power output is not possible. Otherwise, an RTG system for power has no effect on control of the spacecraft as described in Section 5.

10.4 SPINNERS

A spin stabilized spacecraft provides a rather different setting for attitude control. Generally, such a craft needs only one set of sensors, sensing one celestial reference, to maintain control of two axes. If these sensors are mounted on the spin platform of the spacecraft, the output rate of these sensors provides, in addition, a measure of the spin rate of the platform.

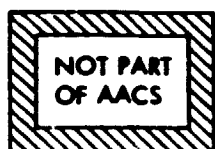
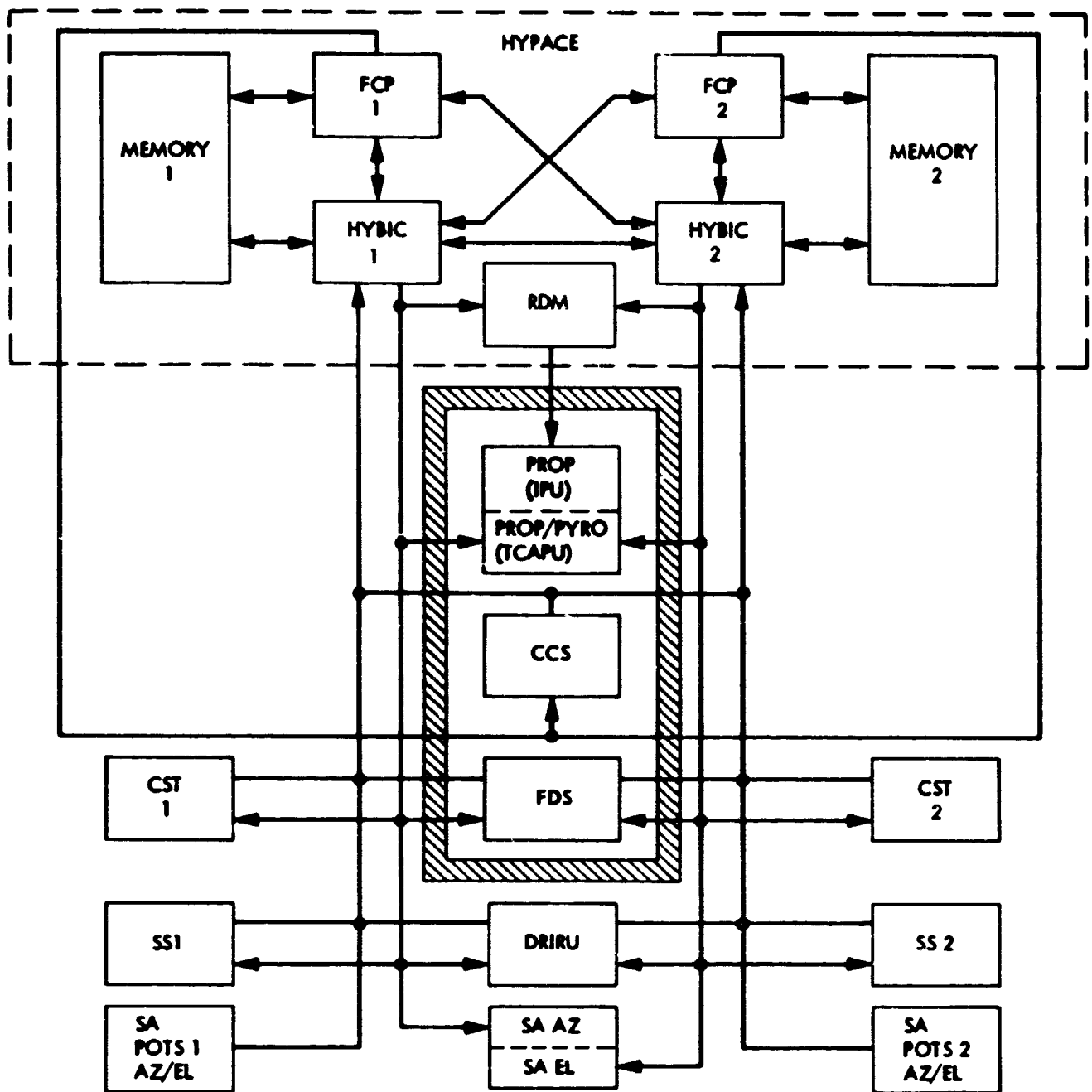


Figure B1-8. Voyager AACs Functional Block Diagram

As a result, a spin stabilized spacecraft requires a much simpler reference loss and reacquisition algorithm. The acquisition of 2XR may proceed as indicated in the initial acquisition and 2XR reacquisition routines of Section 5. However, 1XR acquisition may merely mean timing the 2XR sensors to provide rate information for the spin platform. Hence, a 1XR reacquisition may never be needed except as part of a 2XR reacquisition procedure.

In addition, with sensors mounted on the spin platform, such a spacecraft likely can use only one celestial reference. Hence, a notion of a redundant reference scheme is not possible.

SECTION 11

MISCELLANEOUS

This paragraph is devoted to a few thoughts about the algorithms and model presented in the preceding paragraphs.

The notions of celestial reference and reference scheme are developed in a fashion supporting several generalized alternatives. A generalization of celestial reference may be a reference derived from a signal generated by another satellite. This could be a particularly effective reference for an earth orbiter, if a Global Positioning Satellite (GPS) system were available. A pair of reference schemes may be independent to the extent that only one celestial reference is needed for 3-axis control. DSCS III has a version of this with its all-axis Sun control. Axial control need not be directly dependent on fine celestial sensors. DSCS III infers yaw control from the roll sun sensors.

Failure modes need not be the terminal states of the spacecraft modes of operation. In several scenarios a failure mode was established in preference to a swap of processor circuitry. This was not the case in Voyager, where mission requirements demanded that an attempt be made to obtain full 3-axis control. Such a closed loop approach at fault correction, with the inherent possibility of endless sequences of swapping circuitry and processors in response to an unplanned fault, was avoided in this algorithm for reasons of simplicity and circumstance. It is anticipated that this model and algorithm could be used for an earth orbiter, where downlink transmission and ground recovery are possible, if the spacecraft maintains some form of attitude control and positioning.

It is also conceivable that a restart capability will be placed on-board. Hence, in the case of the establishment of a failure mode, an initial acquisition sequence may be attempted periodically on the theory that 'glitches' through the circuitry may have caused the problems leading to the establishment of a failure mode. This would particularly be the case if radiation effects lead to a two-reference failure mode. In addition, a

one-reference or one-axis degrade mode may have been established because a circuitry swap was inhibited. Payload considerations may have been a cause. Hence, another try at three-axis control may be attempted at times of presumed low usage of payload functions.

Details of the processing during the active transient timer were not given in the algorithm as described in Section 5. Such a routine encompasses processing which would necessarily be sensor specific. Detailed descriptions of the actual sensors were avoided as much as possible. However, it should be mentioned that in such a transient timer routine a decision to switch from one fine sensor to another must be made. In addition, certain actions should be taken, such as flyback and sweep for a star tracker. These actions fall under the category of validating the fault of a given fine sensor. Also, before a redundant fine sensor can be switched in for control use, a calibration of the sensor must take place. Such a calibration may involve measurements against gyros, or a series of tests with output compared to past or idealized sensor performance criteria. The timing and logical interplay of these processes, within certain pointing and mission-specific constraints for attitude control, seem best suited for a mission application.

Appendix B
Section B2
Power Load Fault Management

DESCRIPTION OF AUTONOMOUS FUNCTION

SECTION 1

FUNCTION NAME: POWER LOAD FAULT MANAGEMENT

SECTION 2

FUNCTIONAL DESCRIPTION

The load fault management routine provides a means for the spacecraft (S/C) to automatically detect and recover from a failure in an electrical load on the power subsystem. A spacecraft load failure will be assumed by this routine whenever a particular load impedance moves outside of a predetermined range for the device being monitored.

2.1 Figure B2-1 shows the functional block diagram for a single load.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

3.1.1 Payload Support

The spacecraft shall be maintained in a state such that it is capable of providing support to the payload functions. No single point failure mode of any element, including software, shall cause the loss of support to the payload function.

3.2 SPACECRAFT REQUIREMENTS

3.2.1 Primary Power

The spacecraft primary power bus shall be maintained in a state such that it is capable of providing power to the user load. No single point failure mode of any component shall cause the loss of primary power.

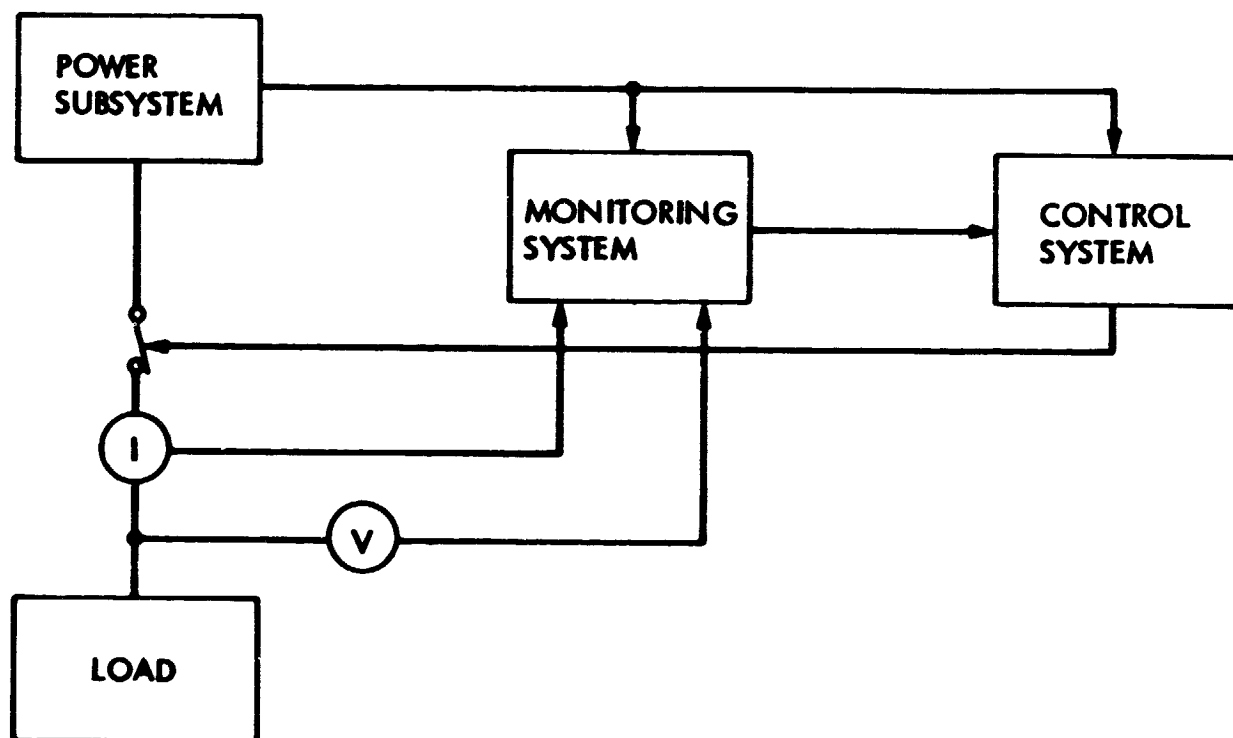


Figure B2-1. Load Fault Management Functional Block Diagram

SECTION 4

SUBSYSTEM FUNCTIONAL DESCRIPTION

4.1 DRIVING REQUIREMENTS

4.1.1 Load Fault Monitoring

The power subsystem shall require spacecraft load impedance monitoring to determine load fault conditions. Currently, S/C loads are fuse-protected, mostly unmonitored, and not under direct power control. Fuses have only one-shot capability and are unreliable at best. Information as to the condition of the load (go or no go) is derivable only through telemetry data, from which it is sometimes difficult to ascertain a load's status.

In order to alleviate the above problems, a Load Fault Management (LFM) system is required. The block diagrams of Figure B2-2 show a comparison between the functions of a general fault detection system and a basic LFM system.

4.2 PURPOSE

4.2.1 Design Parameters

The LFM system shall be designed to 1) increase power system reliability, 2) reduce or eliminate the dependence on fuses, 3) maintain up-to-date impedance values of all loads, 4) provide immediate load replacement upon failure of a primary device, 5) make intelligent and autonomous decisions regarding power management, 6) provide capability of keeping degraded loads on line.

4.3 DESCRIPTION

4.3.1 Inputs and Outputs

The I/O would consist primarily of inputs for the load monitoring circuitry, concepts for the control circuitry, and bi-directional lines for communications to the Spacecraft Central Computer. See Figure B2-3.

4.3.2 Monitoring System

Computer measurements are made of load voltages and currents from which impedances are computed. If any load impedance moves out of a predetermined range, corrective action is taken.

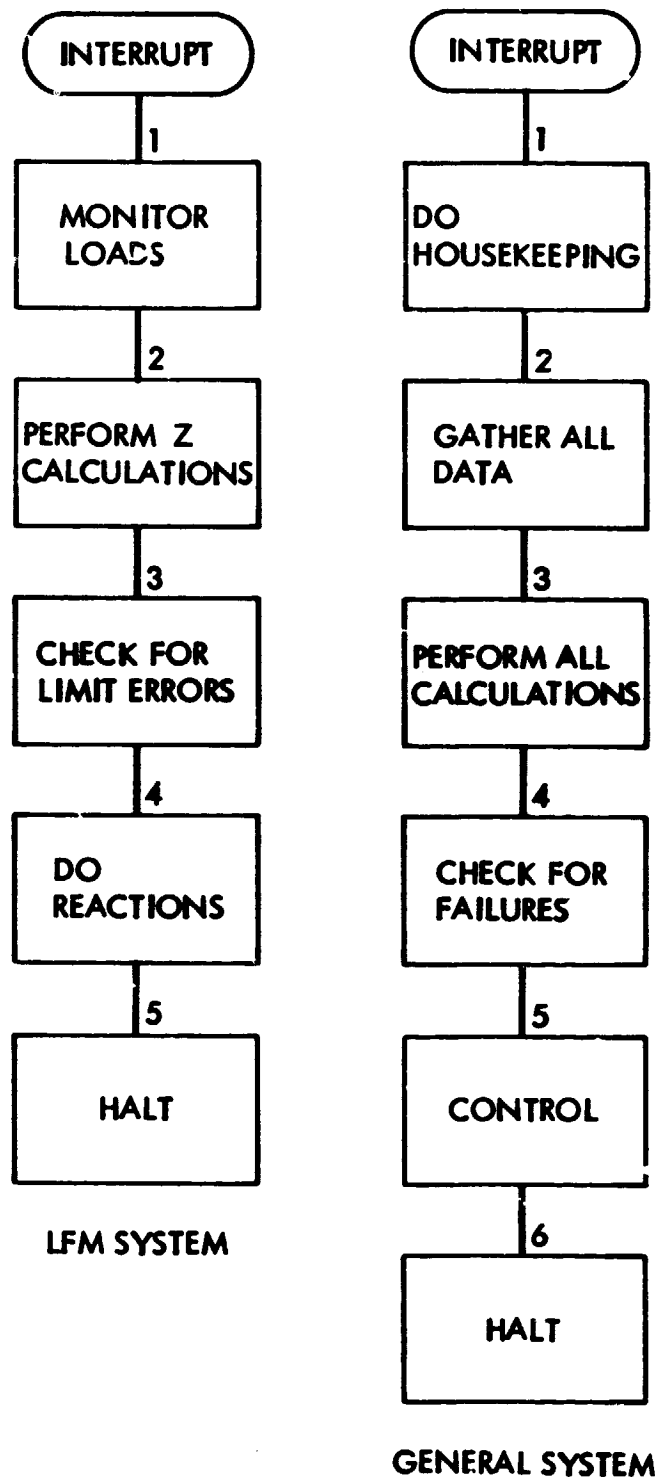


Figure B2-2. Comparison of Basic Functions: General System vs LFM System

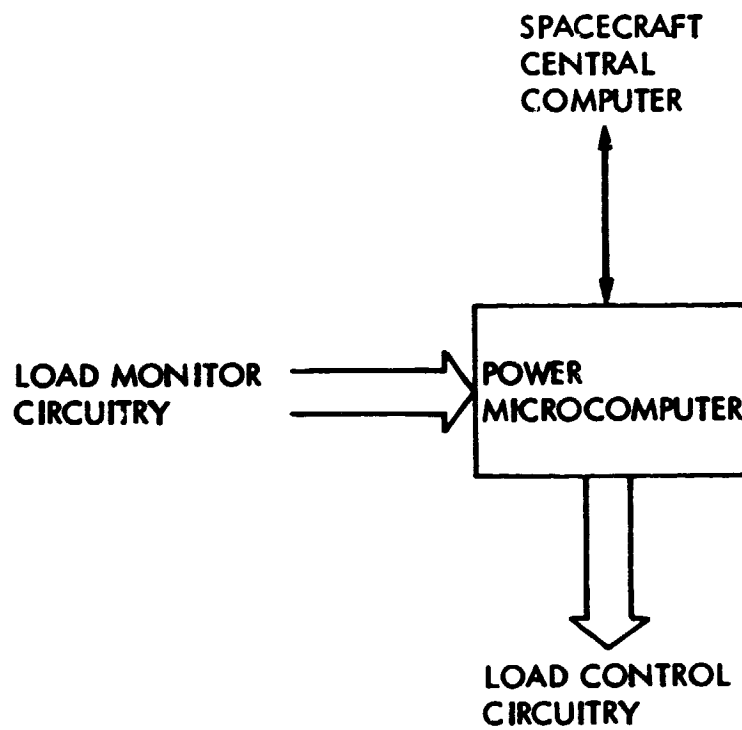


Figure B2-3. Load Fault Management Input/Output Schematic

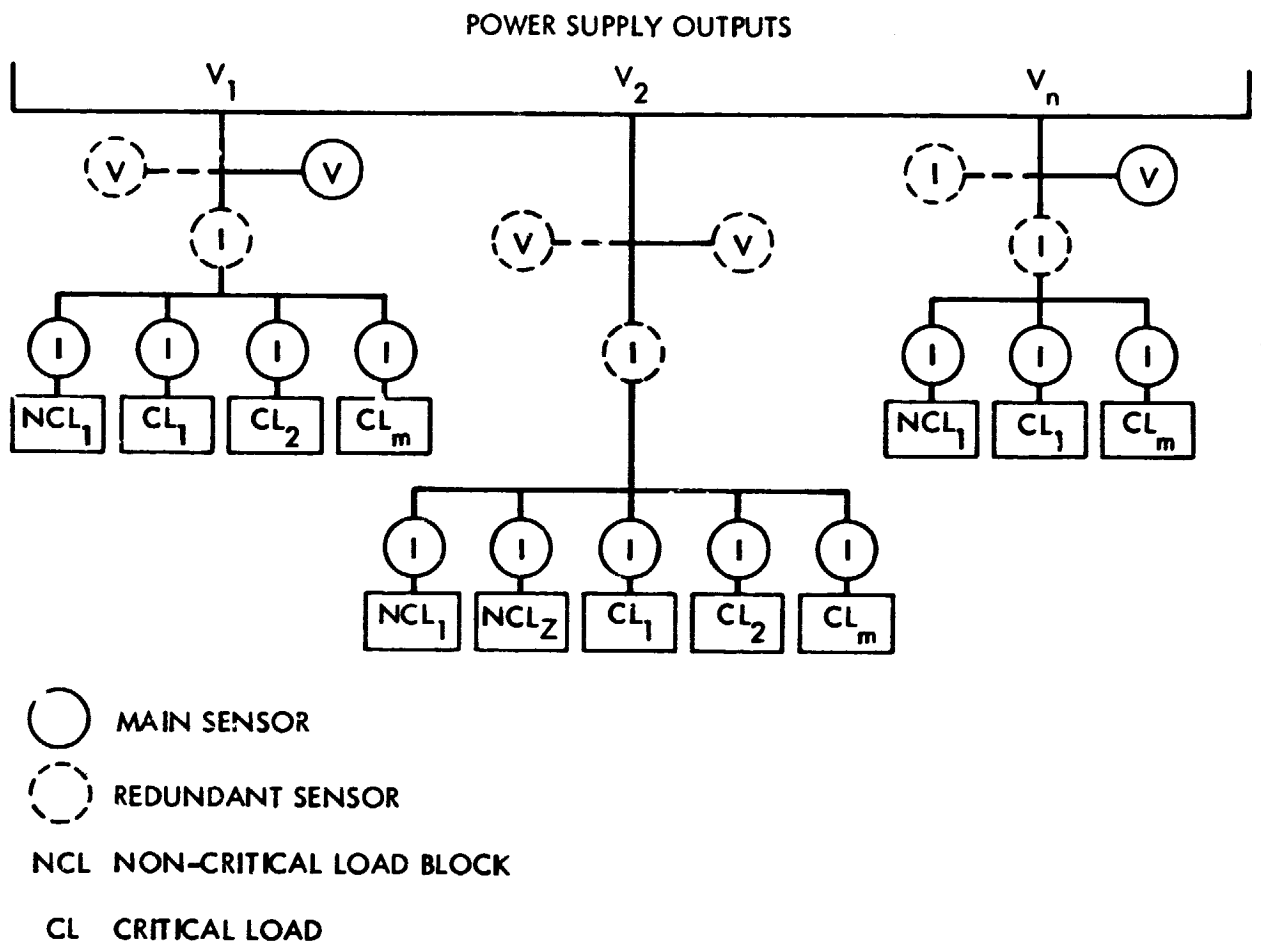


Figure B2-4. Load Monitor Scheme

One voltage sensor must be used for each of n voltage outputs from the power supply. See Figure B2-4. In addition, a second voltage sensor should be used on each voltage output for redundancy and fault verification.

Loads are divided into two groups: critical and non-critical. For example, a critical load might be the ACS computer while a non-critical load might be a heater. Each of the critical loads has a current sensor on its input. The non-critical loads are grouped into one or more blocks with one current sensor at the input to each block. Efficient and inexpensive redundancy can be obtained by placing a final current sensor on each voltage output of the power supply.

4.3.3 Correction System

When the monitoring system detects a failed device, the device's ID is passed to the Correction Routine. The Correction Routine then follows predetermined instructions and removes the faulty load, applies a backup, changes operating parameters, etc., depending on the device and mission requirements.

As in the monitoring system, critical loads are separated from non-critical loads. See Figure B2-5.

If the failure occurs in any load, the corrective measures might be to 1) remove the load, 2) apply the back-up, 3) record relay switchings, and 4) set or clear necessary flags. Also, the most recent impedance of the failed load is recorded. Such information may be useful if the back-up device fails, in which case the original unit may be brought back on line if its characteristics are better than those of the backup.

4.3.4 Load Fault Management Algorithm

4.3.4.1 Decision Points. See Figure B2-6, Flowchart 1: Compare impedance (Z_j) with limits. During the design of the loads, the designer will calculate values for maximum and minimum permissible impedances for each load. These values will be entered into a data base as Z_{jmax} and Z_{jmin} and constitute the "limits" referred to in this decision point. The limits may be altered to suit, for example, a particular power management profile or results of test data. The changes may never be allowed to exceed the Z_{jmax} or fall below Z_{jmin} of the specific devices.

See Figure B2-7, Flowchart 2: Any Z-flags set: one bit is checked to see if any impedances are out of limits.

Compare Z with limits: the same as in Flowchart 1.

All Z complete: This is merely a way of preventing an endless loop in the event of an accidental entry into the branch, and is the same as in Flowchart 1.

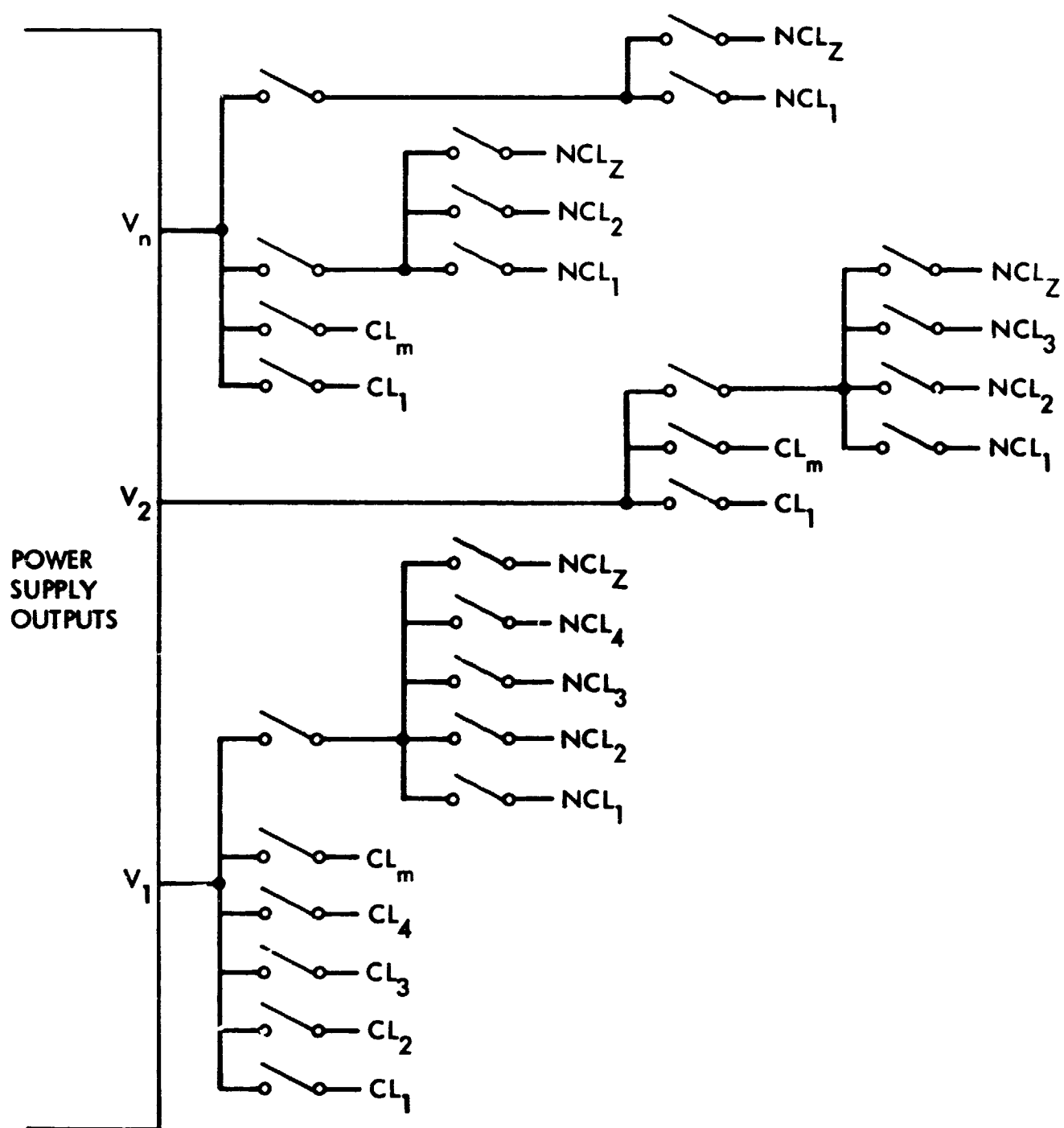


Figure B2-5. Load Control

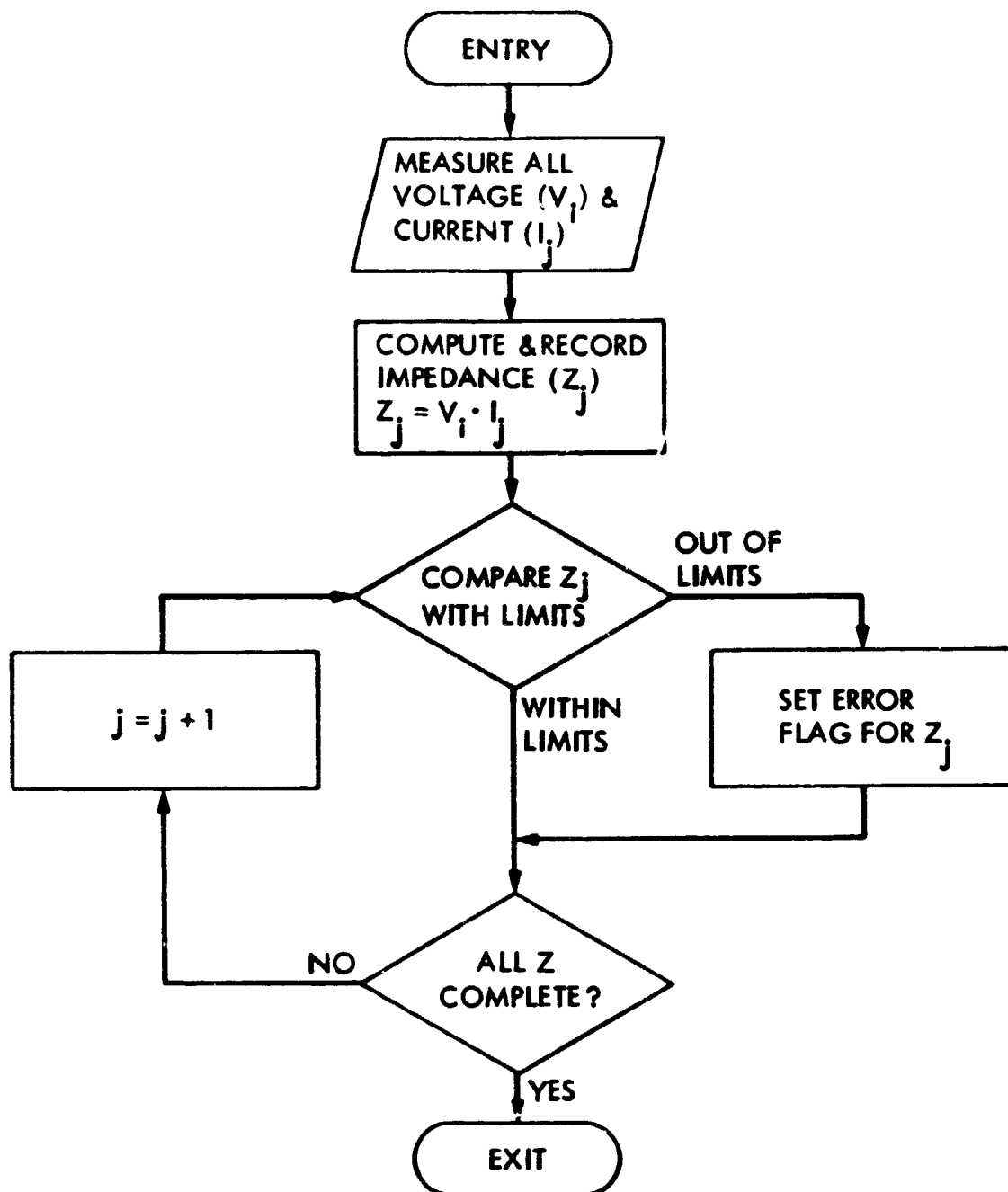


Figure B2-6. Flow Chart 1: Load Monitor

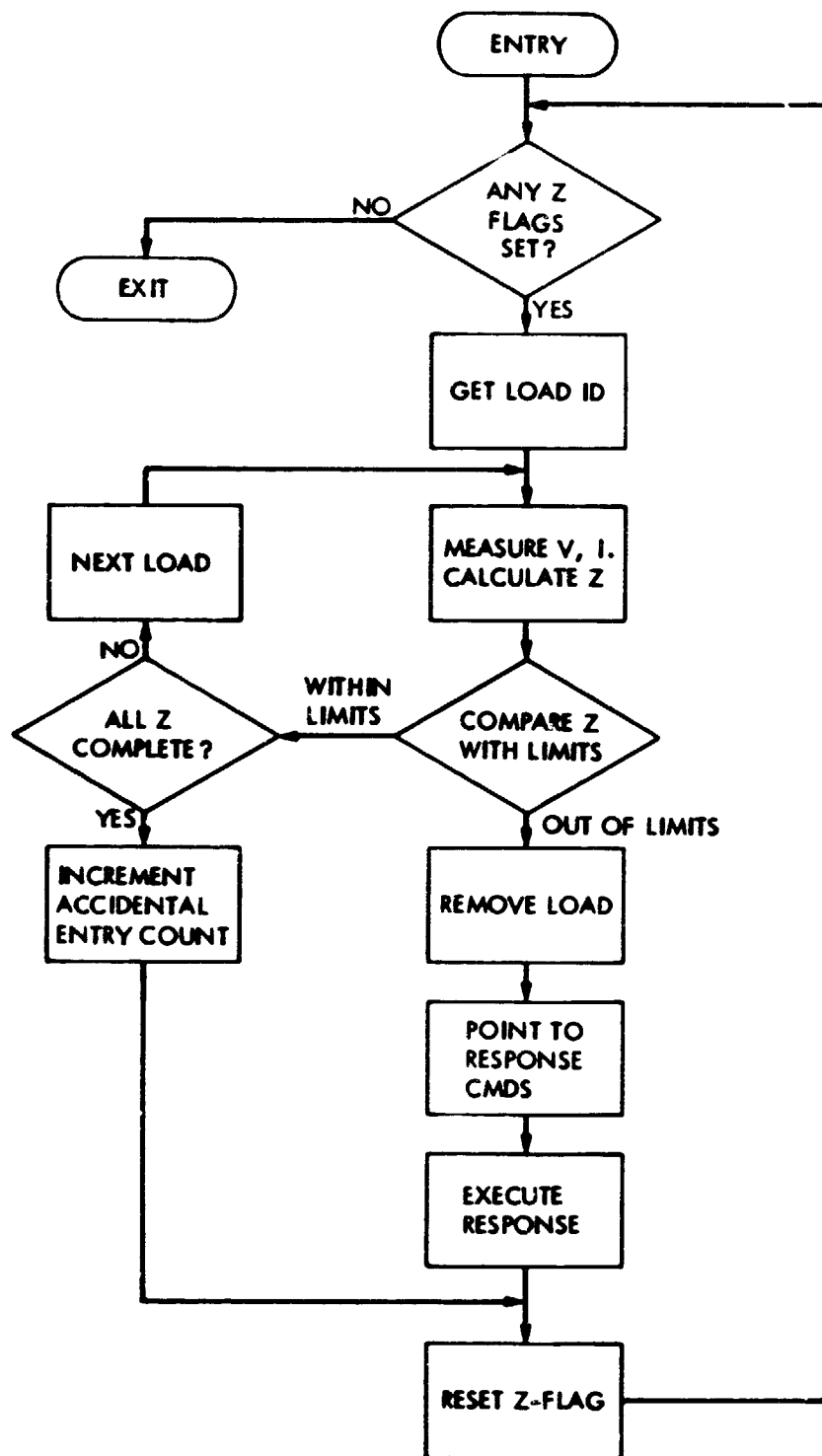


Figure B2-7. Flow Chart 2: Load Failure Isolation and Response

4.3.4.2 Timing. The computer that handles the Load Fault Management would probably be an interrupt-driven system with interrupts every 10 ms. Depending on the size of the tasks, the 10 ms may be altered.

While the routines must be in the order shown in Figure B2-2, other routines and functions will probably be included in the functional blocks.

If there is not enough time in one cycle to accomplish all of the required tasks and if the cycle time has already been stretched to its maximum, the designer may choose to operate the separate routines in different cycles. The necessity of doing this is rather unlikely, though the option does exist.

Appendix B
Section B3
Power Processor Fault Management

DESCRIPTION OF AN AUTONOMOUS FUNCTION

SECTION 1

FUNCTION NAME: POWER PROCESSOR FAULT MANAGEMENT

SECTION 2

FUNCTIONAL DESCRIPTION

The power processor fault management routine provides the capability for a spacecraft to automatically detect and recover from a failure in a power processor. Failure will be detected as a change in efficiency of a power processor outside a predetermined range. This routine will then isolate the failed unit and replace it functionally with a redundant unit. Power processors included DC-DC converters, DC-AC inverters, battery chargers and series regulators.

2.1 FUNCTIONAL BLOCK DIAGRAM

The functional block diagram for a single power processor is shown in Figure B3-1.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

3.1.1 Payload Support

The spacecraft shall be maintained in a state such that it is capable of providing support to the payload functions. No single point failure mode of any element shall cause the loss of support to the payload function.

3.2 SPACECRAFT REQUIREMENTS

3.2.1 Power Processing Functions

The spacecraft power processing functions shall be maintained in a state such that they are capable of providing all of the processed power requirements to the spacecraft and payload throughout the mission. No single point failure mode of any power system element shall cause the loss of primary power.

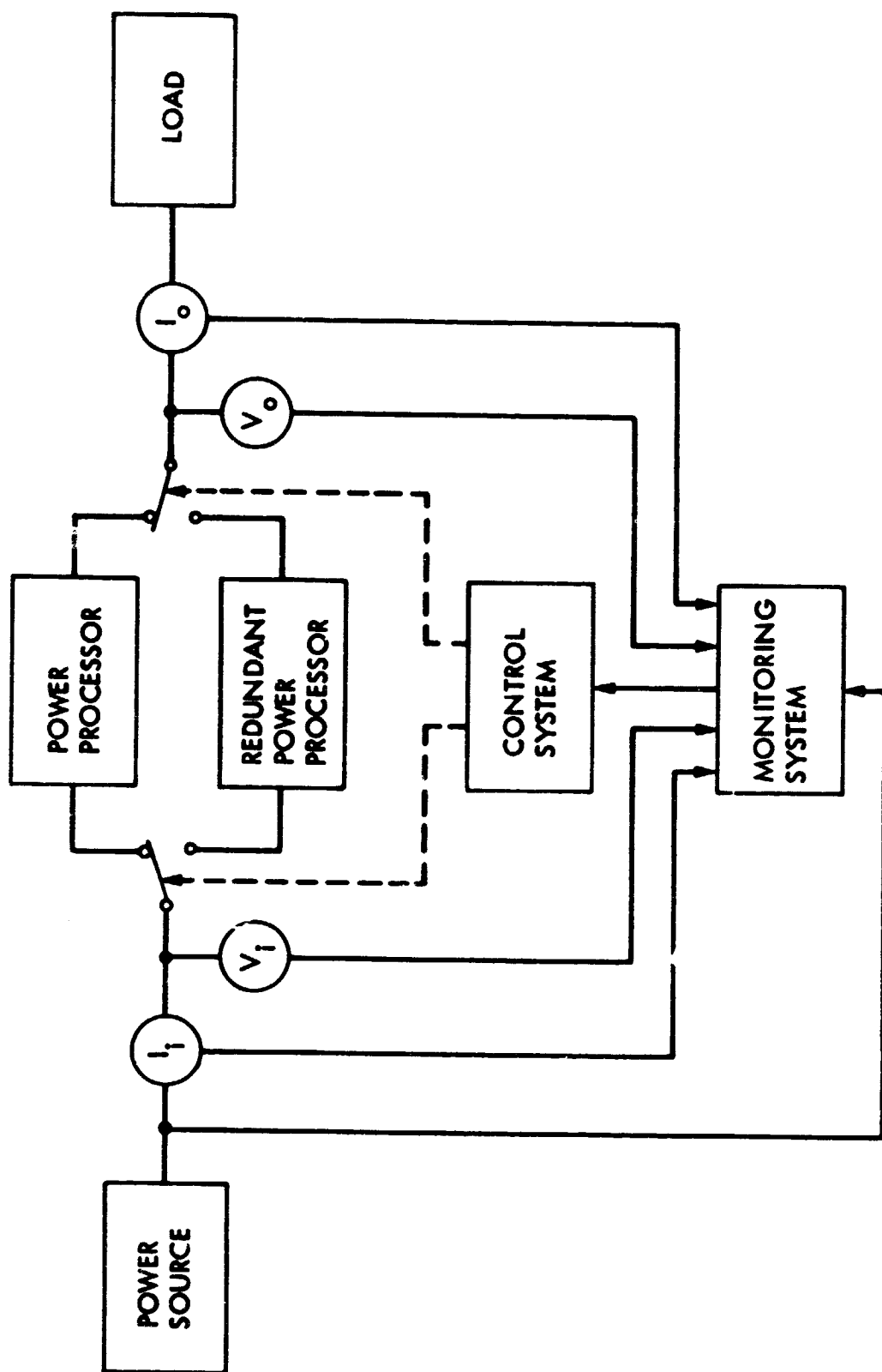


Figure B3-1. Power Processor Fault Management Functional Block Diagram

SECTION 4

SUBSYSTEM FUNCTIONAL DESCRIPTION

4.1 DRIVING REQUIREMENTS

Efficiency calculations are required to be performed on each active power processor to detect degradation or failure. Several methods are currently used to isolate a failed power processor. These methods include fusing, relay disconnect based on out-of-tolerance output voltage, and hardware logic disconnect of the power bus (non-essential power bus) which feeds several loads including the failed processor.

Not only are fuses one-shot devices but they are ordinarily selected with a rating much higher (> 2 times) than the maximum anticipated current. This means that only a severe overload will cause a quick disconnect of the failed unit. Hardware logic disconnect of a failed processor does not provide flexibility, i.e., battery charger output voltage varies over a wide range during normal operation. Disconnecting a multi-load power bus due to a single load failure usually requires extensive ground segment analysis to isolate the fault and implement recovery action.

The Power Processing Fault Management system eliminates the above problems and provides the flexibility to accommodate planned or unplanned mission operation changes.

4.2 PURPOSE

Power Processing Fault Management System shall be designed to 1) increase power system reliability, 2) eliminate the one-shot feature of fuses, 3) maintain up-to-date efficiency values on all power processors, 4) provide immediate replacement of failed primary processors, 5) provide the capability for maintaining a degraded processor on-line.

4.3 DESCRIPTION

4.3.1 Inputs/Outputs

The inputs to the Power Processing Fault Management System include voltage and current measurements of the input and output of the power processor, updates on stored efficiency limits from the spacecraft central computer and processor on/off status. The outputs include updates on the processor efficiency, control signals to the processor switching relays and primary/redundant processor status.

4.3.2 Monitoring Function

As shown in Figure B3-2, Flowchart 1, the power processor input and output voltages and currents are monitored on a periodic basis. The efficiency of the power processor is computed and compared to a stored limit value. If the calculated value is outside the stored limit value, recovery action is initiated.

There are three key requirements for the monitoring function; 1) it must not respond to normal transients, 2) sampling and computations must be performed with sufficient frequency that recovery can be accomplished before disrupting other spacecraft functions, 3) the recovery routine should include a fault verification sequence.

4.3.3 Recovery Function

When the monitoring function detects a failed processor, the recovery function performs a sequence of instructions to either verify the fault, or detect a failed transducer.

If the processor failure is verified, power is switched from the failed unit to the redundant processor. If a transducer failure is detected, the appropriate flag is set and communicated to the spacecraft central computer.

4.3.4 Power Processor Fault Management Algorithm

4.3.4.1 Decision Points. Input and output power parameters are sampled on each power processor. The efficiency of each one is computed and compared to its corresponding limit values. If the efficiency is within limits, the computed value is used to update a designated memory location for access by the spacecraft central computer. If the efficiency is not within limits, the computed value is stored in a fault value location with a time tag, to provide an audit trail. The load fault management routine is then implemented to insure that a faulted load is not causing an erroneous efficiency computation.

If the efficiency remains out of limits with no load faults, the recovery function is initiated, as shown in Figure B3-3 (Flowchart 2).

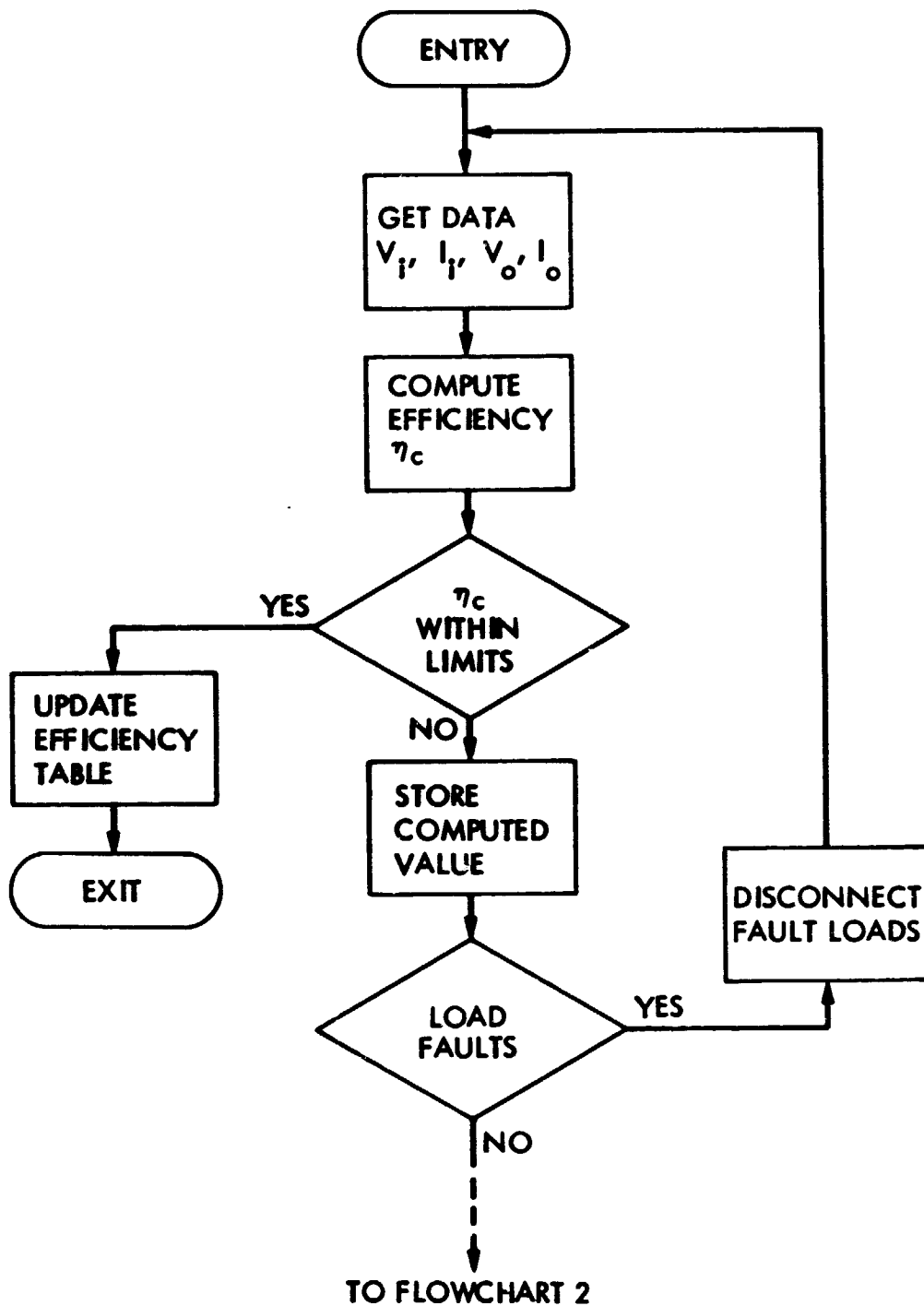


Figure B3-2. Flow Chart 1: Power Processor Monitoring Function

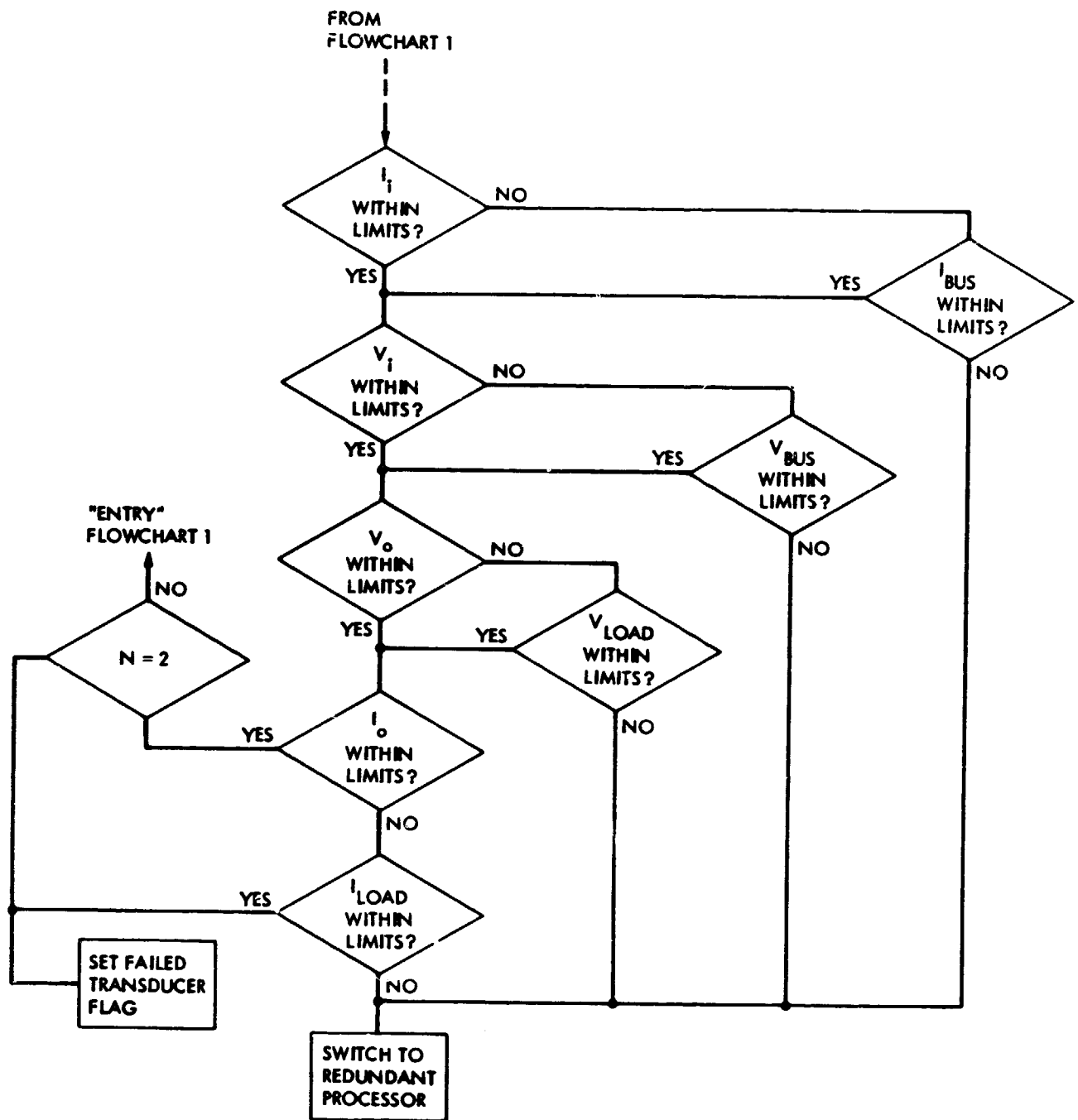


Figure B3-3. Flow Chart 2: Power Processor Recovery Function

The four parameters (V_i , I_i , V_o , I_o) used to compute efficiency are sequentially compared to limit values to verify a fault condition. The most important fault parameter, in terms of the effect on the other spacecraft functions, is the power processor input current. This parameter is tested first. An out-of-limit result is verified by testing a redundant or related current measurement. If the out-of-limit condition is verified, a redundant power processor is switched in. If the input current is within limits, the input voltage parameter is tested, primarily to detect a failed transducer, since it would be improbable that the input current would remain within limits and the input voltage out of limits. Verification of an out-of-limit input voltage is performed on a redundant or related transducer, and if true causes a switch to the redundant processor. If no out-of-limit condition has been detected and verified, the sequence continues in a similar manner for the output voltage and current parameters. If the output current is within limits, the monitoring function is repeated to determine if the initial out-of-tolerance efficiency indication was due to a transient condition. If the same results are obtained (no verified out-of-limit condition), it is assumed that one of the transducers has failed and a designated flag is set and communicated to the spacecraft central computer.

4.3.4.2 Timing. Based on utilizing an interrupt driven microcomputer system (IDMS) similar to the successful Automated Power System Management (APSM) breadboard, no timing problems are anticipated. In addition most power systems provide sufficient energy storage, i.e., batteries and capacitors, to support a fault condition for at least several seconds.

SECTION 5

SUBSYSTEM FUNCTIONAL REQUIREMENTS

5.1 INPUTS

Power processors input and output currents and voltages, on/off status indicators, updated limit values.

5.2 PROCESSING

Computation of efficiencies and comparison of stored data.

5.3 OUTPUTS

Commands to processor switching relays. Status and computational results to the spacecraft central computer.

SECTION 6

INTERFACE LIST MATRIX

TBD

SECTION 7

PERFORMANCE REQUIREMENTS

TBD

SECTION 8

IMPLEMENTATION CONSIDERATIONS

The Power Processor Fault Management approach described herein assumes a processor with a single output. Additional analysis is required to develop an algorithm to perform the same function on a multiple output processor, but without the complexity of performing an analysis on each output. Implementation of an audit trail, and its configuration, will be dependent on the specific mission requirements.

8.1 SOFTWARE

Power Processor Fault Management is a software routine executed on a repetitive basis by a microcomputer.

8.2 HARDWARE

All control circuitry and primary sensing circuits are normally contained within the power system. Secondary sensors, necessary to provide verification data, may be located in other subsystems.

The decision to incorporate a dedicated microcomputer in the power system is considered spacecraft specific. Tradeoffs are necessary which include factors such as quantity and timing of computation and control functions, number of added interfaces (wiring, connectors) required to provide power system data to a multi-use computer and the capability of a multi-use computer to perform with two or more independent fault status inputs.

8.3 ESTIMATED RESOURCES

Resource requirements for this task are spacecraft specific for the power system electrical configuration, i.e., quantity of redundant power processors. Ground support manpower for performance monitoring, failure

analysis and response determination would be reduced by implementation of this function.

SECTION 9

VALIDATION REQUIREMENTS

9.1 ANALYSIS REQUIREMENTS

TBD

9.2 TEST REQUIREMENTS

TBD

Appendix B
Section B4
Battery Cell Replacement

DESCRIPTION OF AN AUTONOMOUS FUNCTION

SECTION 1

FUNCTION NAME: BATTERY CELL REPLACEMENT

SECTION 2

FUNCTIONAL DESCRIPTION

The Battery Cell Replacement routine provides the capability for recovering the full use of a battery which has been degraded by one or more failed cells. This routine addresses the problem of 1 to N cells (where N is a number much less than the total number of cells in the battery) failing as a result of abnormal degradation or a random event.

2.1 FUNCTIONAL BLOCK DIAGRAM

The functional block diagram is shown in Figure B4-1.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

3.1.1 Payload Support

The spacecraft shall be maintained in a state such that it is capable of providing support to the payload functions. No single point failure mode of any element shall cause the loss of support to the payload function.

3.2 SPACECRAFT REQUIREMENTS

3.2.1 Storage Function

The spacecraft electrical energy storage function shall be maintained in a state such that sufficient energy storage is available to provide all planned spacecraft and payload requirements with a minimum additional margin of (TBD%). No single point failure mode of any energy storage element shall cause the loss of primary power.

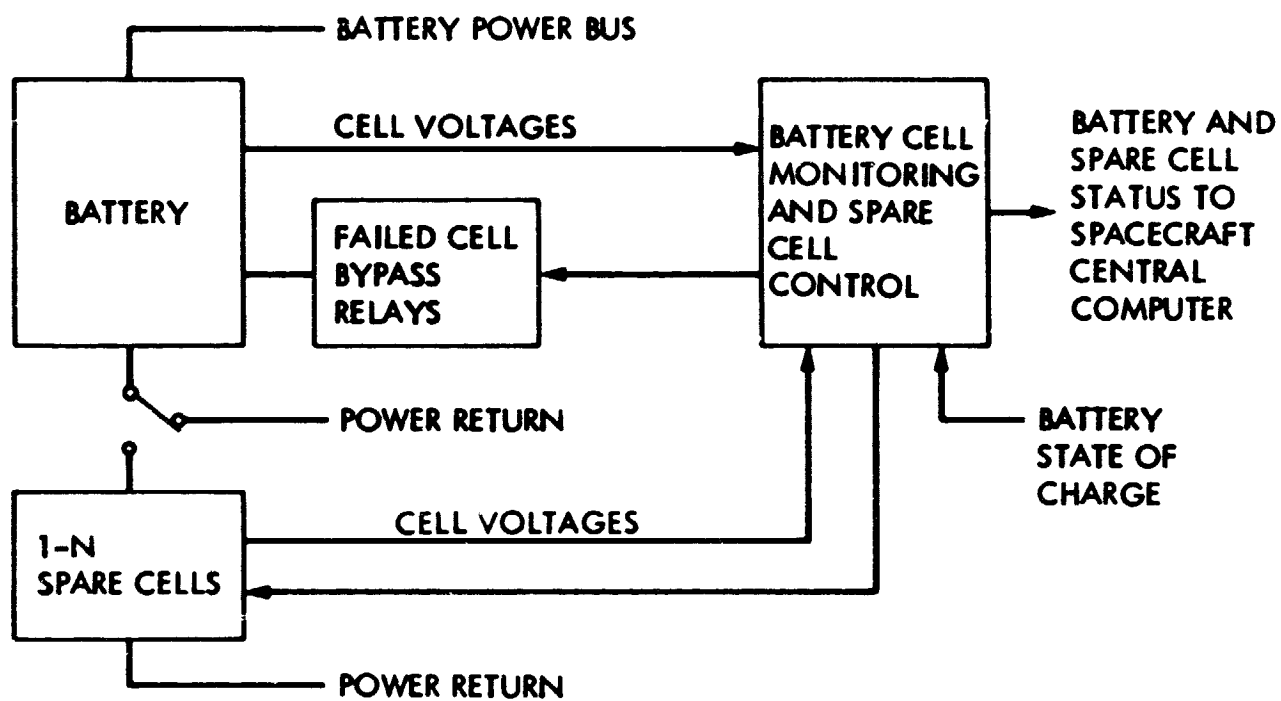


Figure B4-1. Battery Cell Replacement Functional Block Diagram

SECTION 4

SUBSYSTEM FUNCTIONAL DESCRIPTION

4.1 DRIVING REQUIREMENTS

Measurements of individual battery cell voltages are required to identify a specific failed cell. Currently the total battery voltage is monitored to detect abnormal operation, and total battery redundancy is implemented to achieve the required reliability of the energy storage function.

The capability of replacing a few failed cells can therefore provide a significant mass saving.

4.2 PURPOSE

Battery cell replacement shall be designed to 1) increase power system reliability, 2) minimize power system mass, 3) provide immediate replacement for (TBD) failed cells.

4.3 DESCRIPTION

4.3.1 Inputs/Outputs

The inputs to the Battery Cell Replacement system (see Figure B4-1) include battery cell voltage measurements and battery state of charge. The outputs include failed cell bypass relay commands, spare cell relay commands and status data to the spacecraft central computer.

4.3.2 Monitoring Functions

All battery cell voltages are monitored on a periodic basis. If the battery is not in a reconditioning cycle and the state of charge is above 50%, the cell voltages are compared to a predetermined limit value. The identification number of each cell whose voltage is below the limit is stored in memory. If one or more failed cells are identified, the recovery routine is initiated.

Verification of cell failure is also included in this routine.

4.3.3 Recovery Function

If a failed cell(s) is verified, the recovery function performs a sequence of instructions to either replace the failed cell or remove the failed cell and modify the battery charger parameters to operate with the lower cell count battery.

4.3.4 Battery Cell Replacement Algorithms

4.3.4.1 Decision Points. As shown in Figure B4-2, Flowchart 1, each cell voltage is sampled and compared to a predetermined limit. If all cell voltages are within limits, a status flag is set for access by the spacecraft central computer. If one or more failed cells are identified, their identification numbers are stored in memory for access by the recovery routine. To verify a failed cell, the battery voltage is compared to a computed limit range.

$$\pm V_{\text{limit}} = (V_c \times N) \pm TPD \text{ (MV)}$$

where V_c = voltage of a non-failed cell

N_c = number of cells in the battery

If the battery voltage is within limits the routine is repeated to determine if there was a data error or a failed transducer. If the battery voltage is not within limits the recovery function is initiated.

Referring to the Figure B4-3, Flowchart 2, if no spare cells are available, the failed cell is located from the stored identification data and electrically disconnected from the battery. The failed cell location in the battery is short circuited to maintain the electrical path. The battery charger maximum voltage and voltage cutoff parameters must be modified for the reduced battery voltage.

If a spare cell is available, the failed cell is removed, its location shorted in the battery, and the spare cell switched in - in series with the battery. The spare cell count is updated and the recovery routine repeated for additional failed cells.

4.3.4.2 Timing. There are no significant timing requirements for this function.

SECTION 5

SUBSYSTEM FUNCTIONAL REQUIREMENTS

5.1 INPUTS

Battery cells and total battery voltages.

5.2 PROCESSING

Comparisons to stored data and computations of battery voltage.

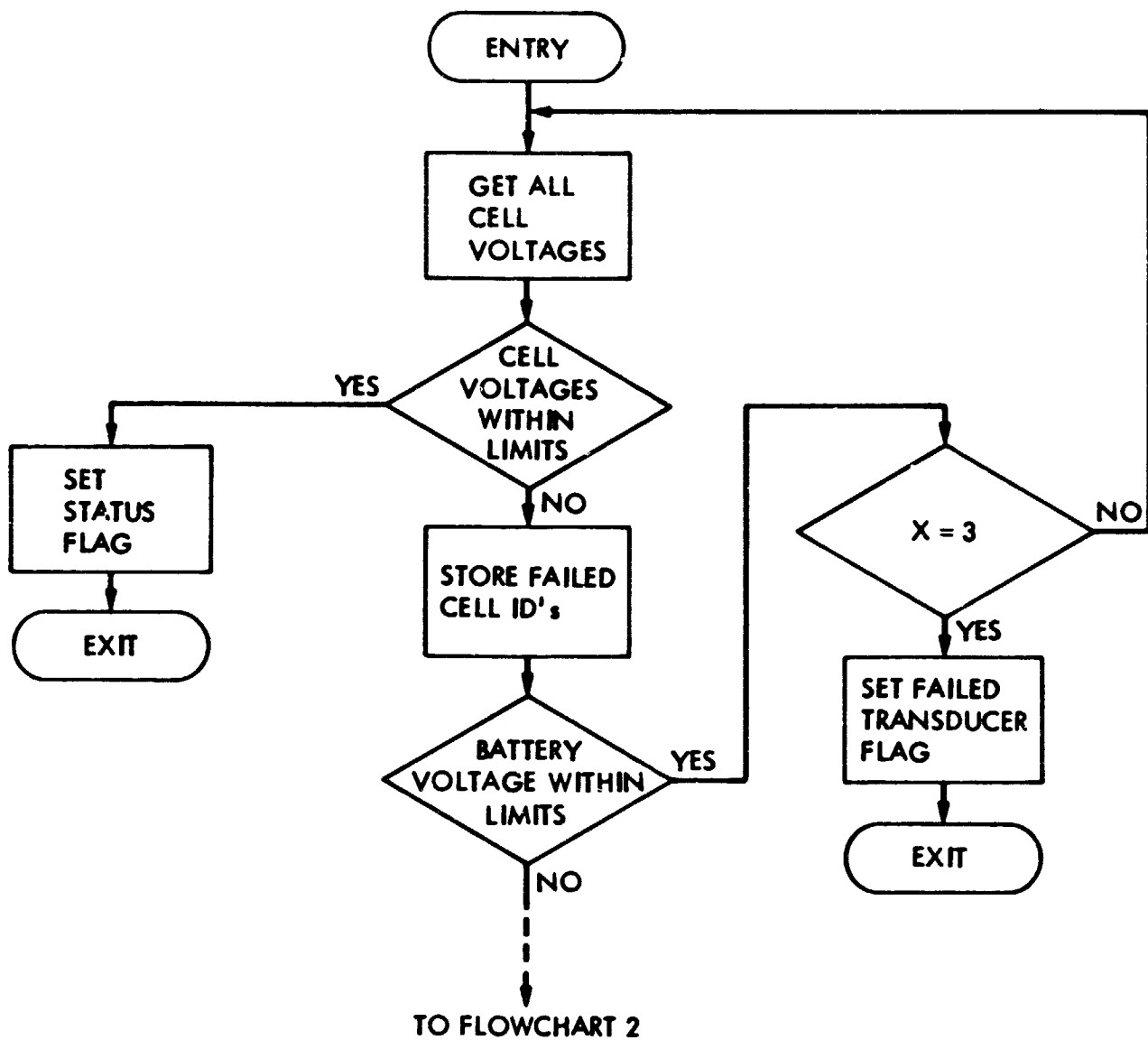


Figure B4-2. Flow Chart 1: Battery Cell Monitoring

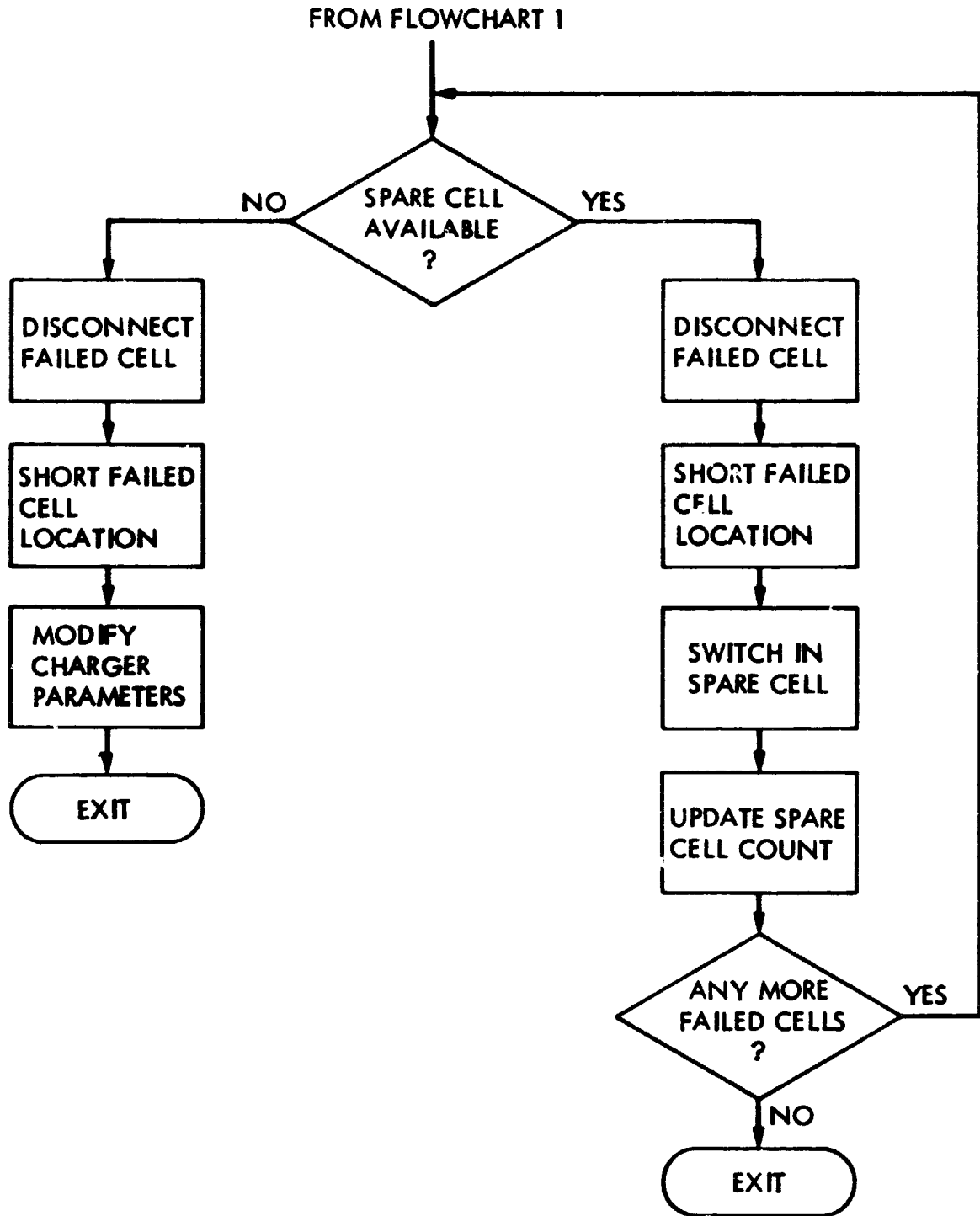


Figure B4-3. Flow Chart 2: Battery Cell Replacement

5.3 OUTPUTS

Commands to battery and spare cell relays. Status and computation results to spacecraft central computer.

SECTION 6

INTERFACE LIST/MATRIX

TBD

SECTION 7

PERFORMANCE REQUIREMENTS

TBD

SECTION 8

IMPLEMENTATION CONSIDERATIONS

The battery spare cell management approach described herein is based on the successful demonstration of the concept on the Automated Power System Management (APSM) breadboard. Application of this approach requires tradeoffs to be performed to optimize the final design. For example: what are the reliability vs complexity tradeoffs for each of the following spares configurations; individual cells, two - one quarter size packs, one-half size battery?

8.1 SOFTWARE

Battery Cell Replacement is a software routine executed periodically by a microcomputer.

8.2 HARDWARE

All battery cell switching relays would be included in the battery housing to minimize line losses.

8.3 ESTIMATED RESOURCES

Resource requirements for this task must include the results of the tradeoff referred to above in Paragraph 8.0.

SECTION 9
VALIDATION REQUIREMENTS

9.1 ANALYSIS REQUIREMENTS

Tradeoffs

TBD

9.2 TEST REQUIREMENTS

TBD

APPENDIX C

EXAMPLE FLIGHT ALGORITHMS FOR AUTONOMOUS CONTROL
AND FAULT MANAGEMENT

TABLE OF CONTENTS

APPENDIX C

	Page No.
SECTION C1: MOIMAU Routine	C1-1
SECTION C2: CMDLOS Routine	C2-1
SECTION C3: CORKER Routine	C3-3
SECTION C4: RFLOSS Routine	C4-1
SECTION C5: Celestial Sensor Fault Detection/Correction	C5-1
SECTION C6: Trajectory Correction and Attitude Propulsion Unit (TCAPU) Fault Detection Routine	C6-1
SECTION C7: TRNSUP Routine	C7-1
SECTION C8: ERROR Routine	C8-1

Appendix C
Section C1
MOIMAI Routine

DESCRIPTION OF AN AUTONOMOUS FUNCTION

SECTION 1

FUNCTION NAME: MOIMAU

SECTION 2

FUNCTIONAL DESCRIPTION

The purpose of the Mars Orbit Insertion (MOI) maneuver power transient routine is to provide a means to continue the Mars Orbit Insertion maneuver sequence activity executed by the Computer Command Subsystem (CCS) in the presence of a spacecraft (S/C) power transient or Attitude Control Subsystem (ACS) Attitude Control Electronics (ACE) power changeover. The ACE power changeover is an indication that the prime ACE unit has detected a failure and might not be able to control the S/C attitude.

2.1 Figure C1-1 illustrates the MOIMAU inputs, processing and outputs.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

The basic requirement is to achieve Mars orbit. The criticality of the MOI propulsive maneuver is such that all means available shall be used to insure its proper execution.

3.2 SPACECRAFT REQUIREMENTS

In support of the basic mission requirement stated above in Paragraph 3.1, the following shall be implemented:

3.2.1 Health Verification

The ability to verify basic health of one half of the CCS to perform ACS recovery and continue the MOI maneuver sequence.

3.2.2 ACS Reinitialization

The ability to reinitialize ACS to a known state consistent with the ongoing MOI maneuver.

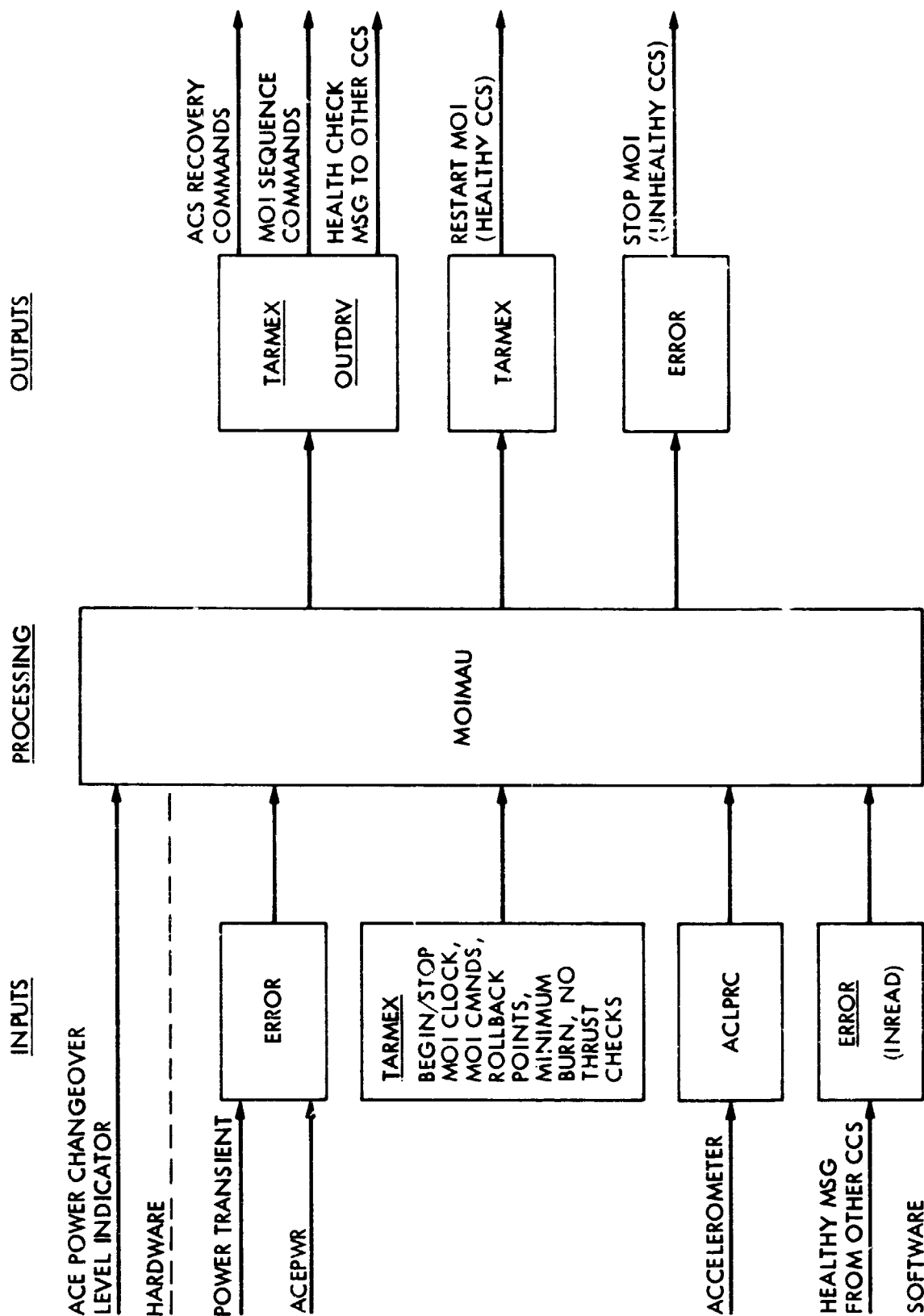


Figure C1-1. MOIMAU Interface Block Diagram

3.2.3 Switching Ability

The ability to switch the standby redundant ACE online in the event of an ACE power changeover.

3.2.4 MOI Restart Capability

The ability to restart the MOI maneuver at established points in the sequence after entry into the ERROR routine (which terminates all sequence activity).

3.2.5 Minimum Motor Burn

The ability to insure a minimum motor burn duration.

3.2.6 Post-MOI Delay

The ability to delay the post-MOI motor burn sequence activity (unwind maneuver) until a 'no thrust' condition exists.

4.0 SUBSYSTEM FUNCTIONAL OPERATIONS

The MOIMAU routine is used only once to execute the spacecraft propulsive maneuver sequence required for Mars Orbit Insertion. MOIMAU is loaded into CCS memory at launch, used at orbit insertion and removed for orbital operations. The maneuver sequence is loaded into CCS memory two months prior to orbit insertion, expended and removed.

4.1 PARALLEL SEQUENCE OPERATION

The maneuver sequence is executed by each half of the CCS in a totally parallel manner to provide the maximum probability that the sequence events will occur.

4.2 SEQUENCE ROLLBACK

Each ACS command of the maneuver sequence issued by CCS is established as a rollback point to which the sequence can be restarted.

4.3 MINIMUM MOTOR BURN CHECK

This function operates to insure that a minimum S/C ΔV is achieved either under primary control of the ACS accelerometer or as timed by the CCS clock.

4.4. NO THRUST CHECK

This function operates to insure that the propulsive motor has shut down before the remainder of the maneuver sequence continues to acquire proper orientation and celestial references.

4.5 OTHER PROCESSOR HEALTH VERIFICATION

This function operates to insure that the healthiest CCS processor will execute the ACS recovery and maneuver restart once the MOIMAU routine is entered due to a S/C power transient or ACE power changeover.

4.6 ACS RECOVERY

This function operates to reinitialize the ACS to a known state consistent with the ongoing maneuver activity and to switch in the standby block redundant ACE if the ACE power changeover has occurred.

4.7 MOI POWER FAIL RECOVERY

This process serves to restart the MOI sequence activity at the last ACS command in the maneuver sequence which was established as the rollback point.

4.8 FUNCTIONAL DATA FLOW DIAGRAM

The functional data flow diagram is shown in Figures C1-2 and C1-3.

SECTION 5

SUBSYSTEM FUNCTIONAL REQUIREMENTS

5.1 INPUTS

5.1.1 MOIMAU Via TARMEX

Calls to MOIMAU for the maneuver sequence via TARMEX will start and stop the CCS 'back-up' seconds clock, request the execution and establishment of ACS commands as rollback points, and activate the minimum motor burn and 'no thrust' checks.

5.1.2 Inputs from ERROR

The INREAD portion of the ERROR routine fetches and makes available to the MOIMAU routine the 'other processor healthy' message.

MOI POWER TRANSIENT AND
ACS SWITCH ROUTINE
(SHEET 1 OF 2)

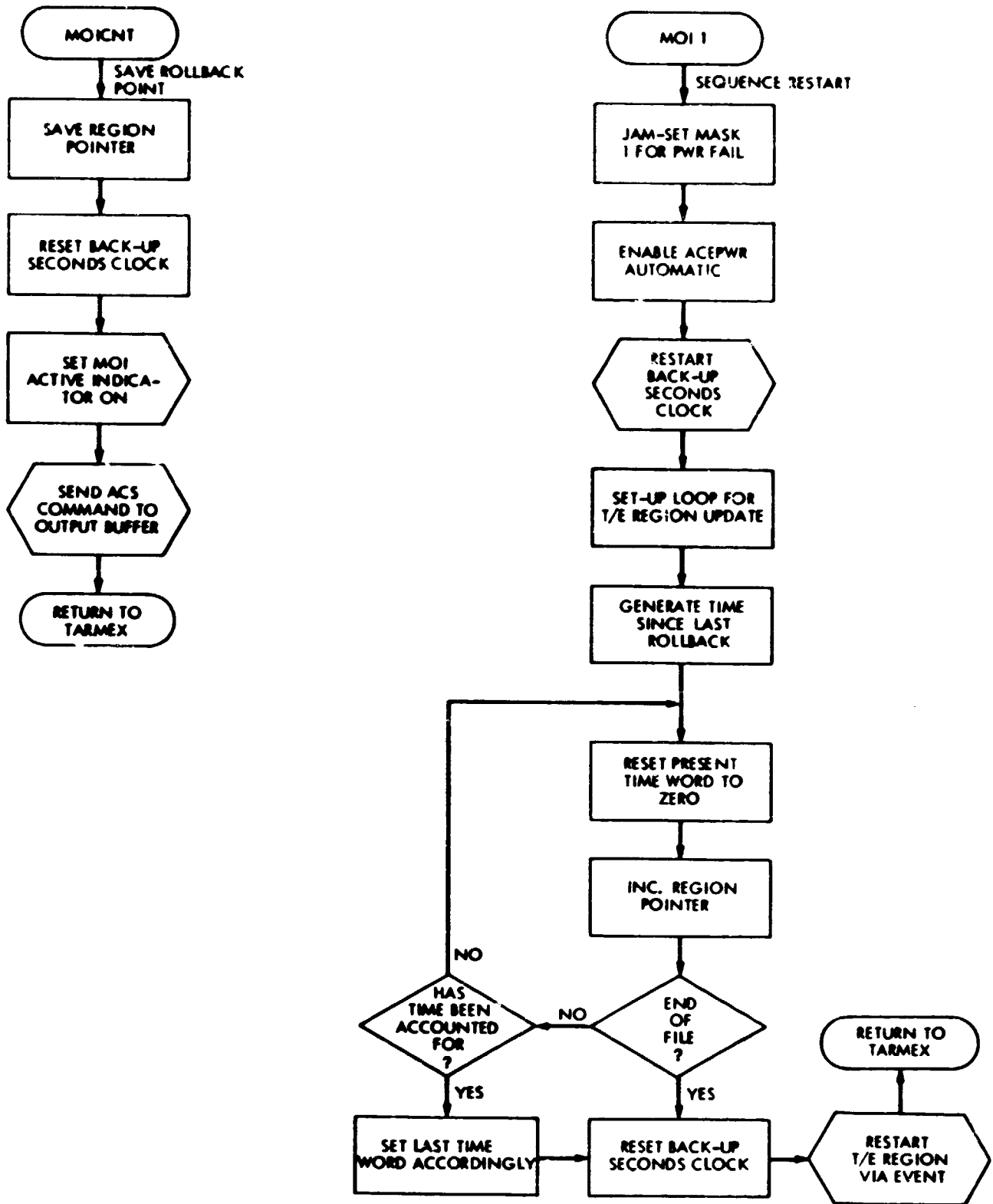


Figure C1-2. MOIMAU Routine Flow Diagram

MOI POWER TRANSIENT
AND ACS SWITCH ROUTINE
SHEET 2 OF 21

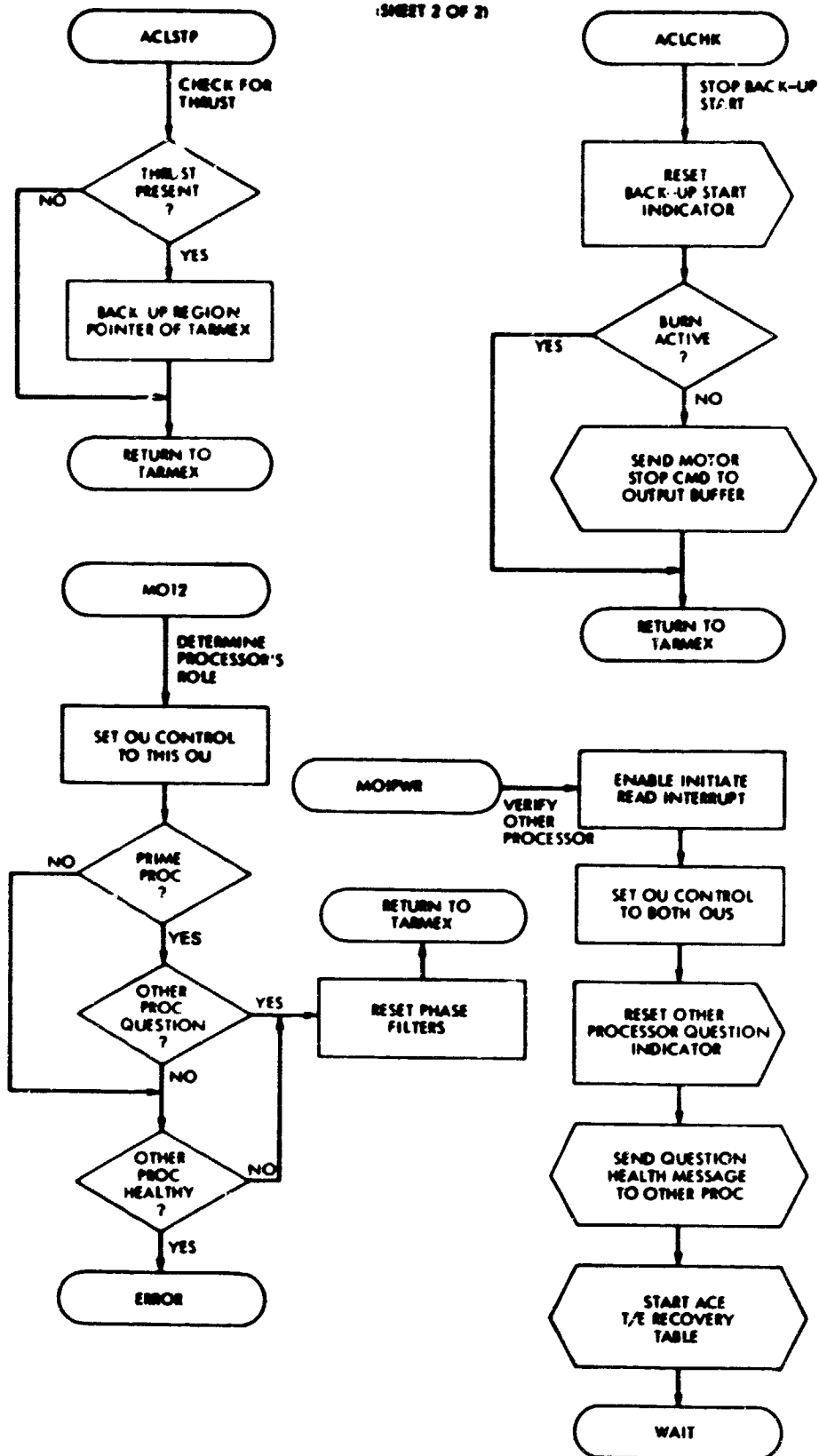


Figure C1-3. MOIMAU Routine Flow Diagram

5.1.3 Accelerometer Routine

The accelerometer routine (ACLPRC) makes available to the MOIMAU routine the 'accelerometer activity' and 'thrust present' indicators.

5.1.4 MOI Restart

A call to the MOIMAU routine from the ERROR routine (which could be entered due to either a CCS hardware-sensed power transient or ACE power changeover) recovers the ACS and restarts the MOI maneuver at the last established rollback point.

5.1.5 CCS Inputs

The routine is also provided with a CCS hardware level input from ACS to indicate a request for ACE power changeover. This is used to switch in the redundant ACE.

5.2 PROCESSING

5.2.1 Initialization

The MOIMAU routine is entered several times during the execution of the MOI maneuver. The first call (MOISEC) occurs before the first sequence rollback point is established and serves to start the 'backup' seconds clock (via TARMEX).

5.2.2 ACS Rollback

After the 'backup' seconds clock is started the maneuver execution proceeds. For each ACS command to be issued in the maneuver sequence, the MOIMAU routine is entered (MOICNT) to establish this point in the sequence as a rollback point. The present sequence point (as generated by TARMEX) is saved, the 'backup' seconds clock (which has been keeping time since the last ACS rollback point), is reset, the ERROR routine call to MOIMAU is enabled, and the ACS command is executed (via OUTDRV).

5.2.3 Minimum Motor Burn Check

This processing works with the accelerometer routine to control the motor burn duration for a specified number of accelerometer pulses or optionally until a minimum motor burn time, as determined by the CCS, occurs. This portion of the MOIMAU routine (ACLCHK) is called once after motor burn start at a point in time which represents the desired minimum motor burn time as clocked by the CCS.

The 'accelerometer active' indicator provided by the accelerometer routine (ACLPRC) is checked to see if an accelerometer controlled burn is still in progress. If there is, the desired S/C ΔV has not been achieved and the ACLPRC routine will terminate the motor burn when the desired number of

accelerometer pulses are input to CCS from ACS. If the 'accelerometer active' indicator shows that the desired S/C ΔV has been achieved or exceeded, the motor burn is stopped at this minimum burn time by executing commands via OUTDRV to close the propulsive engine valve and select roll inertial attitude.

5.2.4 No Thrust Check

This processing works with ACLPRC to check for the absence of thrust after the motor burn stop has been commanded. This portion of the MOIMAU routine (ACLCHK) is called once from the maneuver sequence at a point at least 5 seconds after the expected primary motor burn stop (ACLPRC controlled) to allow the thrust sampling portion of ACLPRC to clear the 'thrust present' indicator. ACLPRC samples for a change in the accelerometer count every five seconds. If this processing indicates thrust has been absent for at least three consecutive sampling periods, then the maneuver sequence proceeds to the next activity (unwind maneuver). If the processing indicates that thrust has not been absent for at least three sampling periods, then the maneuver sequence is held for another 15 seconds and the presence of thrust is checked again for three sampling periods, etc.

5.2.5 Other Processor Health Verification

This is the first process performed upon entry from the ERROR routine. The two halves of the CCS execute the other processor health verification check in a prime/secondary configuration. (This configuration is functionally similar to the master/slave configuration used in the Voyager TRNSUP routine to execute tandem commands. However, the Viking prime vs. secondary processor designation is determined by fixed software indicators in each half of the CCS, whereas the Voyager master vs. slave configuration is determined by the way the sequence is translated on the ground before being loaded into each half of the CCS.) Each CCS half sends a 'health question' message to the other processor, tries to input the other processor's 'health question' message and send a 'processor healthy' message to the other processor. Only one processor, the one which is still functioning and has a faster clock, will actually be able to do this. If the prime processor is able to input the secondary processor's health question message and output a 'processor healthy' message back to the secondary processor it continues on to process the ACS recovery sequence (prime and secondary processor healthy). If the prime processor is not able to input the secondary processor's 'health question' message and output a 'processor healthy' message back to the secondary processor it checks to see if the secondary processor has sent its 'processor healthy' message.

If it has, the prime processor terminates all further activity (secondary healthy, primary less healthy). If the secondary processor has not sent its 'processor healthy' message, the prime processor continues on to process the ACS recovery sequence (secondary dead). If the secondary processor is able to input the 'prime processor healthy' message, it too may be healthy. However, it assumes the prime is more healthy and terminates all further activity (prime healthy, secondary less healthy). If the secondary processor is not able to input the 'prime processor healthy' message, the secondary processor assumes the role of the primary and continues on to process the ACS recovery sequence.

5.2.6 ACS Recovery

The purpose of the ACS recovery sequence is to reinitialize the ACS to a known state consistent with the ongoing maneuver activity. This is done by modifying the recovery sequence out of the maneuver sequence as the sequence activity progresses. The Table C1-1 recovery sequence repeats a total of 9 times, and then makes a check for ACE power changeover. If the ACE power changeover is the reason for entering the routine, the standby block redundant ACE is switched in using Table C1-2, and the ACE recovery sequence again executes 9 times. After this, or if the ACE power changeover was not the reason for entering the routine, MOI power fail recovery is performed.

5.2.7 MOI Power Fail Recovery

After the ACS has been restored and initialized, the back-up seconds clock is restarted and a nominal value of five seconds is used to account for the amount of time which would have been lost with a S/C power transient. TARMEX control of the maneuver sequence is reinitialized with this new time value, then TARMEX once again takes control of the sequence activity. This is required because the TARMEX sequencing activity is terminated by the ERROR routine before entry to the MOIMAU routine.

5.3 OUTPUTS

The MOIMAU routine shall output MOI sequence commands (established as ACS rollback points) via OUTDRV.

The MOIMAU routine shall output ACS recovery command data of Tables C1-1 and C1-2 via TARMEX and OUTDRV.

The MOIMAU routine shall output 'health question' message data to the other processor via OUTDRV.

The MOIMAU routine shall restart the MOI sequence at the established restart point.

The MOI routine shall terminate (via ERROR entry) an unhealthy processor's restart processing.

SECTION 6

INTERFACE LIST

A listing of the external interfaces is shown in Table C1-3.

Table Ci-1. ACS Recovery Sequence

	SEQUENCE AND DESCRIPTION OF COMMAND RESPONSE
1	PREAM PITCH POSITION
2	PREAM YAW POSITION
3	PITCH AND YAW RATE INPUT-GYRO; THRUST VECTOR CONTROL(TVC) GAIN HIGH; TVC ENABLE/INHIBIT; ROLL INERTIAL
4a	PRIOR TO MOTOR BURN START
4b	AFTER MOTOR BURN START
5	INERTIAL REFERENCE UNIT(IRU) 1 ON; IRU 2 OFF; IRU 1 ENABLE; IRU AUTO CONTROL ENABLE
6	INERTIAL/RATE MODE; STOP ROLL TURN; STOP YAW TURN; NEGATIVE TURN POLARITY AND SLCO ENABLE
7a	PRIOR TO YAW TURN START
7b	AFTER YAW TURN START
8	LAUNCH MODE DISABLE; CANOPUS TRACKER(CT) POWER ON; SUN GATE BACKUP RESET; ASOC ENABLE
9	INITIALIZE CANOPUS TRACKER

Table C1-2. ACE Power Changeover Sequence

SEQUENCE AND DESCRIPTION OF COMMAND RESPONSE
1. ATTITUDE CONTROL ELECTRONICS (ACE)-2 SELECT
2. LOW GAIN ANTENNA (LGA) SELECT

Table Cl-3. Interface List, MOIMAU Routine (Sheet 1 of 2)

FROM/TO	WHAT	HOW	REFERENCE
CCS HARDWARE/MOIMAU	ACE POWER CHANGEOVER LEVEL INDICATOR	HARDWARE LOGIC IN CCS	PARA. 5.1; FIG. 2.1-1
ERROR/MOIMAU	S/C POWER TRANSIENT	ERROR EXECUTION	PARA. 5.1; FIG. 2.1-1
ERROR/MOIMAU	ACEPWR ROUTINE ENTRY	ERROR EXECUTION	PARA. 5.1; FIG. 2.1-1
ERROR (INREAD)/MOIMAU	HEALTH CHECK MESSAGE INDICATOR	ERROR EXECUTION	PARA. 5.1; FIG. 2.1-1
ACLPRC/MOIMAU	ACCELEROMETER ACTIVITY INDICATOR	ACLPRC EXECUTION	PARA. 5.1; FIG. 2.1-1
ACLPRC/MOIMAU	THRUST PRESENT INDICATOR	ACLPRC EXECUTION	PARA. 5.1; FIG. 2.1-1
TARMEX/MOIMAU	BEGIN/STOP MOI BACKUP SECONDS CLOCK	MOI SEQUENCE ACTION	PARA. 5.1; FIG. 2.1-1
TARMEX/MOIMAU	MOI SEQUENCE COMMANDS (ROLLBACK POINTS)	MOI SEQUENCE ACTION	PARA. 5.1; FIG. 2.1-1
TARMEX/MOIMAU	MINIMUM MOTOR BURN CHECK	MOI SEQUENCE ACTION	PARA. 5.1; FIG. 2.1-1
TARMEX/MOIMAU	NO THRUST PRESENT CHECK	MOI SEQUENCE ACTION	PARA. 5.1; FIG. 2.1-1
MOIMAU/TARMEX-OUTDRV	ACS RECOVERY SEQUENCE	MOIMAU EXECUTION	PARA. 5.3; FIG. 2.1-1
MOIMAU/OUTDRV	MOI SEQUENCE COMMANDS (ROLLBACK POINTS)	MOIMAU EXECUTION	PARA. 5.3; FIG. 2.1-1
MOIMAU/OUTDRV	HEALTH CHECK MESSAGE TO OTHER CCS	MOIMAU EXECUTION	PARA. 5.3; FIG. 2.1-1

Table C1-3. Interface List MOIMAU Routine (Sheet 2 of 2)

FROM/TO	WHAT	HOW	REFERENCE
MOIMAU/TARMEX	RESTART MOI SEQUENCE (HEALTHY CCS)	MOIMAU EXECUTION	PARA. 5.3; FIG. 2.1-1
MOIMAU/ERROR	STOP MOI SEQUENCE AND MOI RECOVERY/RESTART (UNHEALTHY CCS)	MOIMAU EXECUTION	PARA. 5.3; FIG. 2.1-1

SECTION 7

PERFORMANCE REQUIREMENTS

No additional requirements.

SECTION 8

IMPLEMENTATION CONSIDERATIONS

8.1 GENERAL CONSIDERATIONS AND RATIONALE

The ERROR routine was designed to terminate any preprogrammed sequence currently active. It was, however, given the ability to optionally activate the MOIMAU routine under the conditions of S/C power transient or ACE power changeover. Therefore, for these failure modes which cause ERROR entry but would not necessarily cause erroneous CCS outputs, the MOIMAU routine was designed to select the most healthy CCS processor half to recover ACS, and restart and complete the critical MOI maneuver sequence to achieve Mars orbit insertion.

8.2 REQUIRED RESOURCES

The MOIMAU routine required 121 memory words in each CCS processor.

8.3 JPL EXPERIENCE

This routine worked in a nominal fashion for MOI on both Viking S/C. The routine was not entered by either of the two possible error conditions.

SECTION 9

VALIDATION AND TEST REQUIREMENTS

Validation and test of the MOIMAU routine shall be accomplished at two levels: subsystem and system.

9.1 SUBSYSTEM VALIDATION AND TEST

At the subsystem level, verification of the coded routines' ability to satisfy the specified requirements shall be accomplished by simulation of the execution of the routine on a test computer rather than the host computer. This shall allow easy variation of the input calls and data to validate proper operation of the routine.

As in any software system, individual routines or software modules are designed to operate in conjunction with other software routines or modules. Consequently, the MOIMAU routine shall utilize the following routines during simulation testing at the subsystem level:

9.1.1 Timing Inputs

TRAPS, IMAC shall provide timing inputs to TARMEX so that it can output the ACS recovery commands via OUTDRV.

9.1.2 ACS Recovery Commands

OUTDRV shall process the ACS recovery commands from TARMEX.

9.1.3 Error Entry

ERROR shall provide the response error entry to MOIMAU.

9.1.4 MOI Sequence Entry

TARMEX shall provide MOI sequence entries to MOIMAU and sequence restart response.

9.1.5 MOIMAU Constraints

GLBCNT shall provide the fixed constraints used by MOIMAU during execution.

9.1.6 Variable Memory

VARABL shall provide the variable or 'scratchpad' memory required by MOIMAU.

9.2 SYSTEM (SPACECRAFT) LEVEL VALIDATION AND TEST

TBD

Appendix C
Section C2
CMDL0S Routine

DESCRIPTION OF AN AUTONOMOUS FUNCTION

SECTION 1

FUNCTION NAME: CMDLOS

SECTION 2

FUNCTIONAL DESCRIPTION

The command loss routine provides means for the spacecraft to automatically correct a failure to receive ground commands.

2.1 FUNCTIONAL BLOCK DIAGRAM

The functional block diagram for CMDLOS is shown in Figure C2-1.

2.2 INTERFACE BLOCK DIAGRAM

Figure C2-2 illustrates the command loss routine inputs, processing and outputs.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

3.1.1 Ground Command

Ground command capability will be provided for: loading computer flight sequences, initiating/arming certain sequences, and backing up critical on-board commands and the capability for updating Viking Lander prior to Viking Lander Capsule (VLC) separation.

3.1.2 Failure Modes

No single failure mode of any component shall cause a catastrophic effect on the mission or in the loss of data from more than one scientific experiment or all engineering data.

No single command shall place the S/C in a state that would result in catastrophic loss of the mission.

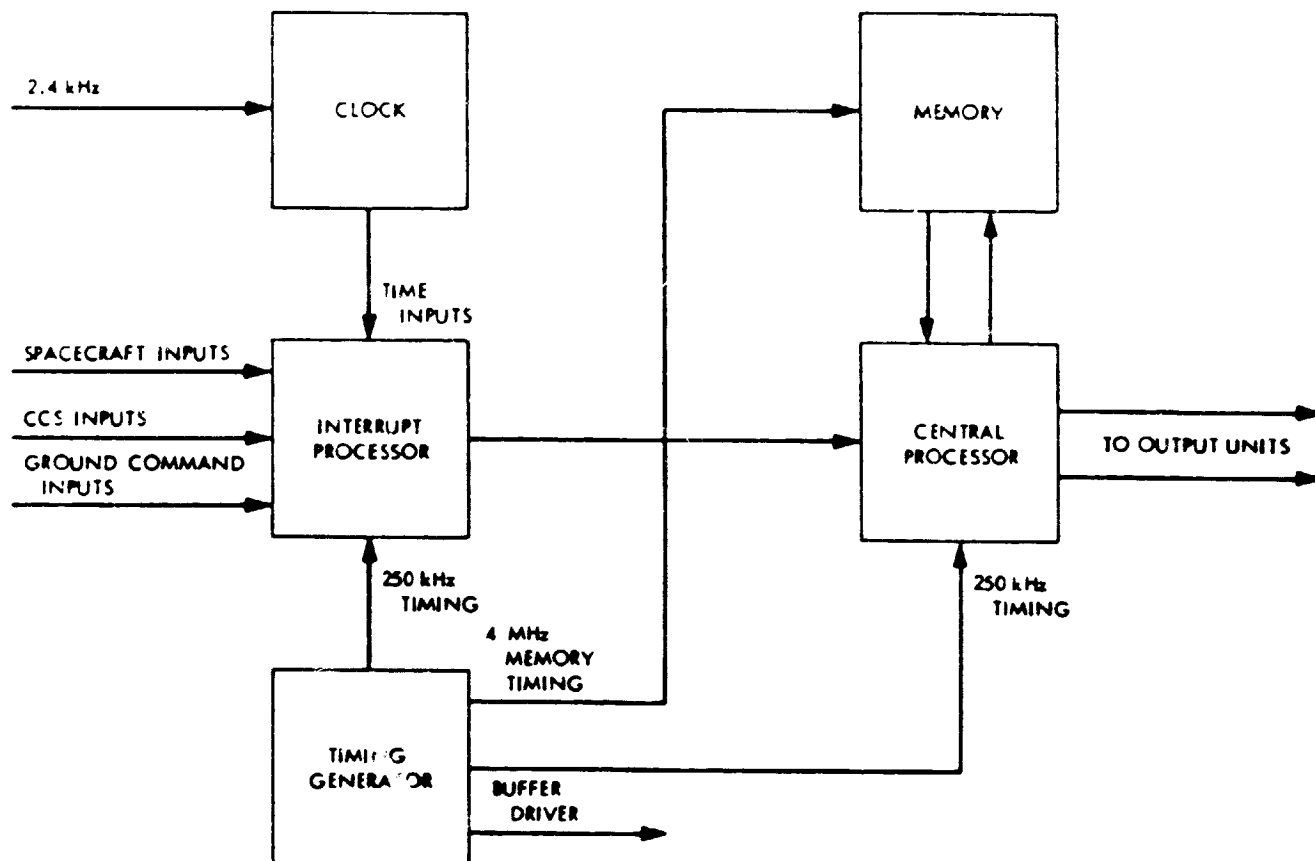


Figure C2-1. Processor Functional Block Diagram

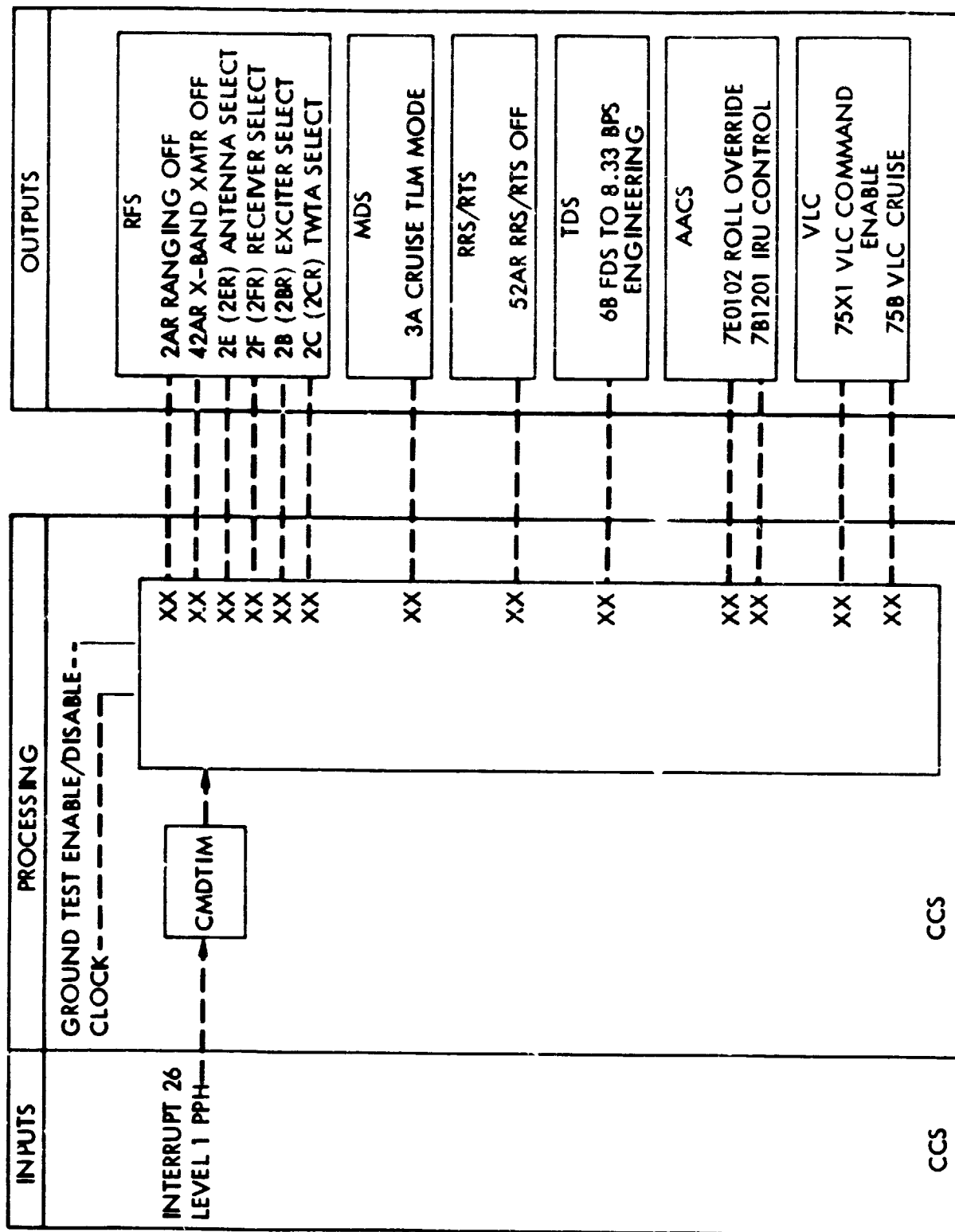


Figure C2-2. C2TLOS Interface Block Diagram

3.1.3 Exit and Access Capability

It shall not be possible to place the S/C in a state such that exit from that state is impossible, nor shall it be possible to cycle the S/C through a state in such a manner that no means of returning to that state is possible.

3.2 SPACECRAFT REQUIREMENT

3.2.1 Priorities

Analysis of mission requirements resulted in the need for significant block redundancy and the formulation of response priorities to direct the design. In order of decreasing priority they are:

- (1) Spacecraft safety and commandability
- (2) Preservation of spacecraft consummables
- (3) Downlink telemetry visibility

3.2.2 Commandability Assurance

To ensure commandability, the spacecraft must exercise all combinations of receivers, command detector units and antennas if ground command is not received during a predetermined amount of time.

3.2.3 Spacecraft Reconfiguration

Since the algorithm must execute during a period when the spacecraft cannot receive and/or decode a command, it is designed to provide complete reconfiguration of the spacecraft's antenna, receiver, and command detector chain.

SECTION 4

SUBSYSTEM FUNCTIONAL REQUIREMENTS

Whenever a specified number of hours have elapsed since the last valid command was received by the CCS, this routine will assume a spacecraft failure and attempt to correct that failure by systematically switching redundant elements until a valid command is received by the Computer Command Subsystem (CCS).

Furthermore, if loss of commandability is due to a false lock condition (a receiver locked up on a downlink spur), the algorithm must also reconfigure the downlink elements (S-Band exciters and traveling wave tube

amplifiers (TWTA's), X-Band XMTR off, Radio Relay Subsystem/Radio Telemetry Subsystem (RRS/RTS) off). To cover all possibilities and be effective for multiple failures, all "uplink" combinations are selected for each "downlink" combination.

Passes through the algorithm continue until a valid command is received.

4.1 FUNCTIONAL DATA FLOW DIAGRAM

The functional data flow diagram is shown in Figure C2-3.

4.2 STATE DIAGRAM

The allowed states are summarized in Tables C2-1 and C2-2.

4.3 LIMITATIONS AND CONSTRAINTS

4.3.1 CMDLOS Restart

When the command loss routine is active it is normally inhibited from being started again until its first activation is completed. Other software (sequence, error, etc.) must not be allowed to remove the restart inhibit before an active command loss routine is finished.

4.3.2 Launch Phase Inhibit

The command loss routine will be inhibited during the launch phase.

SECTION 5

SUBSYSTEM FUNCTIONAL OPERATIONS

5.1 INPUTS

5.1.1 Direct Inputs

(1) CMDLOS entry is from the primary filter/entry linkage.

5.1.2 Indirect Inputs

(1) CMDLOS entry is from negative data in location CMDTIM.

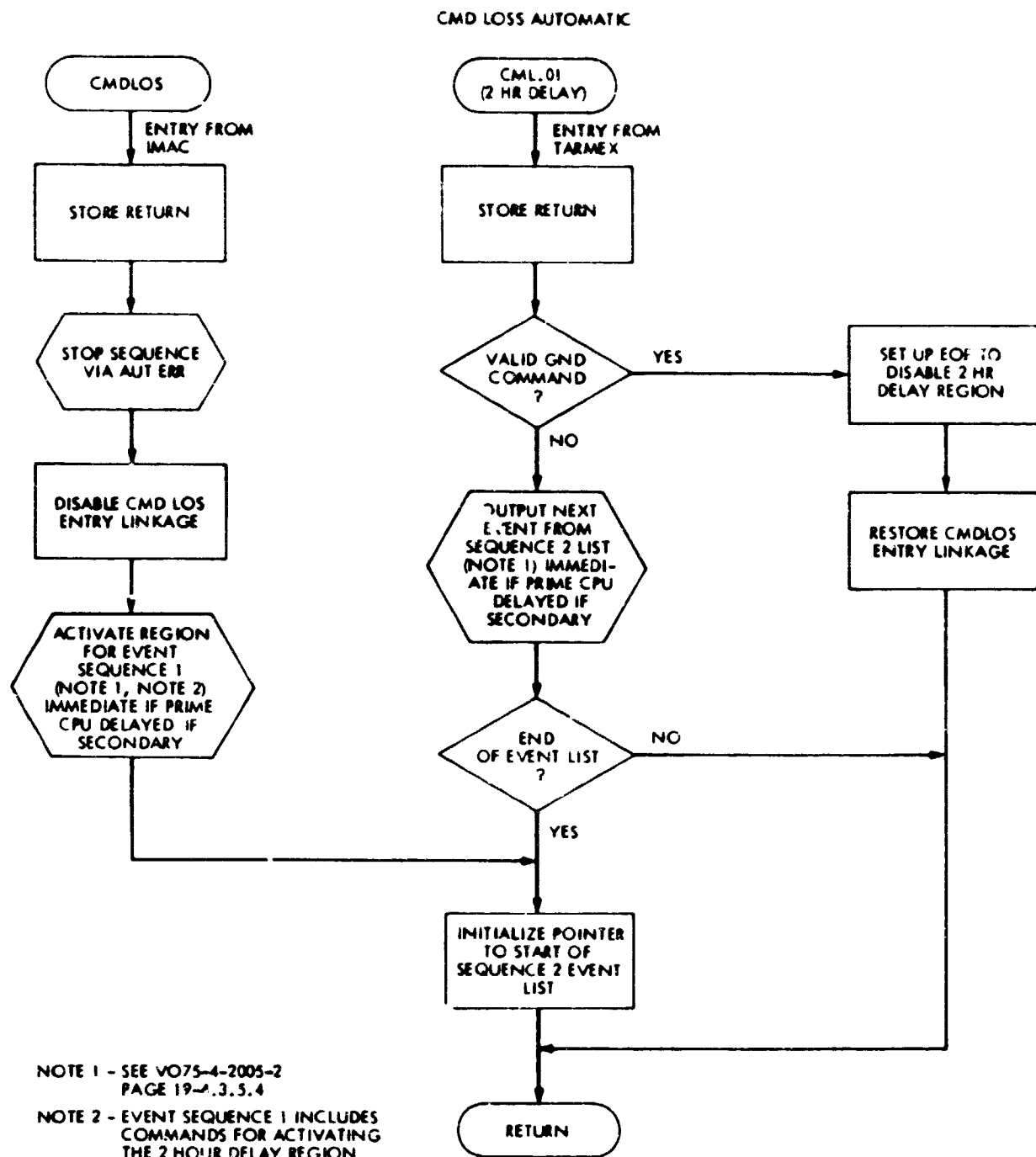


Figure C2-3. CMDLOS Flow Diagram

Table C2-1. Immediate Event Sequence

VIKING
COMMANDS

1.	2 AR	RANGING OFF	DC *
2.	42 AR	X-BAND TRANSMITTER OFF	DC
3.	52 AR	RADIO RELAY SUBSYSTEM/ RELAY TELEMETRY SUBSYSTEM (RRS/RTS) OFF	DC
4.	3 A	CRUISE TELEMETRY (TLM) MODE	DC
5.	6 B	FLIGHT DATA SYSTEM (FDS) TO 8.33 BPS ENGINEERING DATA	CC **
6.	2 ER	LOW GAIN ANTENNA (LGA) SELECT	DC

* DISCRETE COMMAND

** CODED COMMAND

Table C2-2. Delayed Event Sequence

VIKING COMMANDS	PARAMETER DESCRIPTION	COMMAND TYPE
1. 2F (2FR)	RECEIVER SELECT	DC*
2. 2B (2BR)	EXCITER SELECT	DC
3. 2C (2CR)	TRAVELING WAVE TUBE AMPLIFIER (TWTA) SELECT	DC
4. 7B1210 (2E/2ER)	INERTIAL REFERENCE UNIT (IRU) CONTROL/ANTENNA SELECT	CC **/DC
5. 7E0102 (75X1)	ROLL OVERRIDE VIKING LANDER CAPSULE (VLC) COMMAND ENABLE	CC/DC
6. 2FR (2F)	RECEIVE ⁿ SELECT	DC
7. 2BR (2B)	EXCITER SELECT	DC
8. 2CR (2C)	TWTA SELECT	DC
9. 7B1201 (2E/2ER)	IRU CONTROL/ANTENNA SELECT	CC/DC
10. 7E0102 (75B)	ROLL OVERRIDE/VLC CRUISE MODE	CC/CC

* DISCRETE COMMAND

** CODED COMMAND

- (2) CMDLOS entry for prime is from flag indicating primary/secondary Command Processing Unit (CPU).
- (3) CMD.01 entry is the processing activity controlled by sign of data in location CMDTIM.

5.2 PROCESSING

The event timer 'N' is decremented by one each hour, but reset in its initial value each time the CCS successfully receives a command.

If it underflows, CMDLOS is entered.

The response of this routine is the execution of two event sequences: one immediate, and one delayed. The immediate sequence is executed once. The delayed sequence is repeated until a valid ground command is received. Events in the immediate sequence are issued on zero-time centers. Events in the delayed sequence are issued on two-hour centers.

If the Central Processor (CP) is waiting for an interrupt from the Interrupt Processor (IRP), it shall respond to the signal (i.e., program control is transferred to the memory location received from the IRP) within 1 msec.

5.3 OUTPUTS

As shown in Table C2-3, CMDLOS generates output commands to initiate Attitude Control Subsystem (ACS) and Flight Data Subsystem (FDS) mode changes, and hardware switching within the Modulation/Demodulation Subsystem (MDS), Viking Lander Capsule (VLC), Relay Radio Subsystem/Relay Telemetry Subsystem (RRS/RTS), and the Radio Frequency Subsystem (RFS).

SECTION 6

INTERFACE LIST/MATRIX

A listing of the external interfaces is shown in Table C2-4.

SECTION 7

PERFORMANCE REQUIREMENTS

The S/C shall be capable of maintaining a cruise phase operational state for at least 55 hours without ground intervention, except during superior conjunction period, when the requirement shall be for 20 days without ground intervention.

Table C2-3. QMLOS Outputs

VIKING MNEMONIC	PARAMETER DESCRIPTION	COMMAND TYPE
2AR	1 RANGING OFF	DISCRETE
2ER	2 LGA SELECT	DISCRETE
2F (2FR)	3 RECEIVER SELECT	DISCRETE
2B (2BR)	4 EXCITER SELECT	DISCRETE
2C (2CR)	5 TWTA SELECT	DISCRETE
3A	6 CRUISE TLM MODE	DISCRETE
42AR	7 X-BAND XMTR OFF	DISCRETE
52AR	8 RRS/RTS OFF	DISCRETE
6B	9 FDS TO 8 1/3 ENGR	CODED
7B1201	10 IRU CONTROL	CODED
7E0102 (75B)	11 ROLL OVERRIDE/ VLC CRUISE MODE	CODED/ CODED
7E0102 (65x1)	12 ROLL OVERRIDE/ VLC COMMAND ENABLE	CODED/ DISCRETE

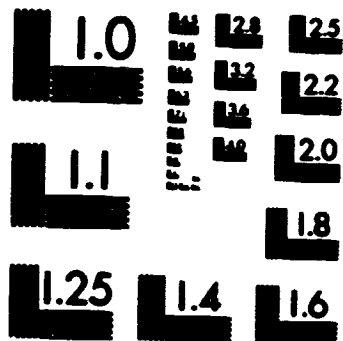
Table C2-4. Interface List

COMMAND LOSS ROUTINE			
FROM/TO	WHAT	HOW (MEDIUM)	INTERFACE DOCUMENT REFERENCE
CMDTIM/CMDLOS	START SIGNAL	'N' HOUR TIMER EXPIRES AND ROUTINE IS ENABLED	PARA. 4; FIG. 2.1-1
CMDLOS/RFS	DC COMMANDS	INTERNAL TO CCS WITH IMMEDIATE & 2 HOUR TIMERS	TABLES 4.2-1 AND 4.2-2 PARA 5.2
CMDLOS/ACS	CC COMMANDS	INTERNAL TO CCS WITH IMMEDIATE & 2 HOUR TIMERS	TABLE 4.2-1 AND 4.2-2
CMDLOS/FDS	CC COMMANDS	INTERNAL TO CCS WITH IMMEDIATE & 2 HOUR TIMERS	TABLE 4.2-1 AND 4.2-2
CMDLOS/RSS/RTS	DC COMMANDS	INTERNAL TO CCS WITH IMMEDIATE & 2 HOUR TIMERS	TABLE 4.2-1 AND 4.2-2
CMDLOS/VLC	DC COMMANDS	INTERNAL TO CCS WITH IMMEDIATE & 2 HOUR TIMERS	TABLE 4.2-1 AND 4.2-2
CMDLOS/MDS	DC COMMANDS	INTERNAL TO CCS WITH IMMEDIATE & 2 HOUR TIMERS	TABLE 4.2-1 AND 4.2-2
MDS/CMDLOS	DC COMMANDS	INTERNAL TO CCS RESET 'N' TIMER	FIGURE 4.1-1
GROUND OPERATIONS/ CMDLOS	1. 'N' TIMER VALUE 2. ENABLE COMMAND	UPLINK COMMAND LAUNCH MODE	

83

6920

3/B



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

SECTION 8

IMPLEMENTATION CONSIDERATIONS

8.1 SOFTWARE

CMDLOS is a software routine executed by the spacecraft's Command Computer Subsystem (CCS).

8.2 HARDWARE

The CCS serving as a central controller is composed of two computers, each of which is used as an interrupt processor, reacting to periodic timing interrupts (hours, seconds, centiseconds, science data frame timing, command bit sync, etc.), and external level interrupts from other subsystems which are typically used to indicate external failures elsewhere in the spacecraft. Both processors have an 18-bit plated-wire (hence nonvolatile) memory containing 4096 words, half of which are "write-protected" such that a "key" must be employed any time this part of the memory is to be altered. Fixed routines for command decoding and failure detection and correction (CMDLOS) are typical of the functions located in write-protected memory. The remaining half of the memory is used to load sequences which control the spacecraft's engineering and science subsystems during trajectory correction maneuvers, science data acquisition and transmittal, and various calibration exercises.

Time intervals chosen for the command loss routine are a function of the ground reaction time to get a valid command received by the CCS.

8.3 ESTIMATED RESOURCES

CMDLOS routine memory storage requirements are 50 words.

8.4 JPL EXPERIENCES

Since the loss of commandability generally precludes any ground-based corrective action, the spacecraft is on its own in providing the needed protection. The only exception to this is the case in which one of the CCS's is unable to process command data it receives from the Command Distribution Unit (CDU). Should this happen, ground control merely needs to reformat the command so that it is executed by the other CCS. By having both CCS's always on-line, receiving and decoding the commands (only command execution need be specified), protection against a single failure resulting in a permanent loss of commandability is provided.

This routine was never activated during the Viking mission.

8.4.1 Flight Modification

No modifications were made to this routine in flight.

8.4.2 Impact of New Technology/Alternate implementation Approaches

An alternate implementation may consider including the CMDTIM tasks (of ascertaining whether valid CMD has been received in the time specified) within the CMDLOS routine.

SECTION 9

VALIDATION

TBD

Appendix C
Section C3
CORKER Routine

DESCRIPTION OF AN AUTONOMOUS FUNCTION

SECTION 1

FUNCTION NAME: AUTOMATIC LEAK CLEARING ROUTINE (CORKER)

SECTION 2

FUNCTIONAL DESCRIPTION

CORKER was the second of two routines developed during the Viking Orbiter (VO) Extended Mission to cope with leaking jet valves in the attitude control gas system. The first, AUTO CORK, was developed for VO-2 to clear roll jet valve leaks. Corker was developed for VO-1 and designed to clear leaks in any axis. Clearing was accomplished by actuating the leaking valve.

CORKER functioned by monitoring limit cycle data obtained via DECOM (Low Rate Engineering Decommulator Executive). Deviations in position error signal which produced deadband hangoff were used as indicators of a leaking jet valve. The action taken to clear the leak depended on the attitude control mode of the spacecraft at the time. If the ACS status word indicated the orbiter was celestially acquired, CORKER commanded the all-axes inertial mode. The gyro-on transient resulted in gas jets actuating in all three axes. If the spacecraft was already in the inertial mode, derived rate was disabled resulting in a position error signal. This caused a limit cycle traverse which resulted in actuating the offending valve.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

CORKER was developed during the Viking Orbiter Extended Mission. No prime mission requirements were applicable. The extended mission requirement was to extend Orbiter life by minimizing attitude control gas consumption. Limited manpower and tracking time during the extended mission required on-board leak clearing to cope with increasing leakage problems on the spacecraft.

3.2 SPACECRAFT REQUIREMENTS

CORKER was designed to clear leaks in any of the twelve attitude control gas jet valves. The routine functioned to clear leaks regardless of the attitude control mode the orbiter was in at the time (celestial or inertial). CORKER was designed specifically for VO-1, which had both inertial reference units (IRUs) working. Therefore, IRU-1 was reserved for operational use, and IRU-2 was dedicated to CORKER for leak clearing attempts.

SECTION 4

FUNCTIONAL OPERATION

The Viking Orbiter attitude control subsystem (ACS) used inert gas supplied by redundant pressure-regulated systems to provide three-axis stabilization. Nitrogen was the primary gas, although during the extended mission residual helium from the propulsion subsystem was supplied to increase spacecraft lifetime. Gas jet leakage became a major problem during the extended mission as gas loss increased, while limited manpower and tracking time reduced the ability of the ground to react effectively. Techniques to autonomously detect and clear leaks were developed on VO-2, where the problem first became severe. Based on the experience obtained using the VO-2 routine called AUTO CORK, a more comprehensive routine, called CORKER, was implemented on VO-1. All leak clearing techniques were predicated on actuating the leaking valve, since particulates were suspected of causing the leaks.

CORKER functioned by monitoring ACS status and limit cycle telemetry using the DECOM routine. If the spacecraft was determined to be celestially acquired, the Canopus tracker (roll) and cruise sun sensor (pitch and yaw) position signals were monitored. With the spacecraft in derived rate (a necessary condition for CORKER to function), a leaking valve would drive the spacecraft across the deadband causing the opposing valve to pulse. When fed through the derived rate circuit into the switching amplifier, the pulses caused a shift in the deadband away from the leaking valve and produced a deadband hangoff proportional to the leak rate. If the offset exceeded limits set in CORKER, normally 2 DN (data numbers) for 5 minutes, IRU-2 was turned on (IRU-1 was reserved for normal spacecraft operational use). Oscillations in gyro capture loop voltages as the gyros attempted to pull in the rate sensing mechanisms resulted in jet valve firings in all three axes. Gyros were turned off after one minute.

If the spacecraft was on inertial references, the leak was cleared by disabling derived rate for 5 minutes. This caused the jet which had been pulsing to remain on until the position error plus rate entered the deadband. This resulted in a rate sufficient to cause the spacecraft to cross the deadband and actuate the leaking valve. Limits used to identify leaks (2 DN for 5 minutes) were normally the same for celestial and inertial modes. In all cases the spacecraft was returned to its initial state after a leak clearing exercise.

Status of leak clearing attempts was tracked by three processor telemetry words. CLKW1 and CLKW2 registered the number of times CORKER issued the derived rate and IRU-2 on commands. These words also recorded which valve caused the commands to be issued. A third word, OPVER, recorded whether CORKER was enabled or disabled, active or inactive and the ACS status word. Normally the processor words were issued once an hour. However, when CORKER was active the appropriate count word was issued immediately.

4.1 FUNCTIONAL FLOW DIAGRAM

Functional flow diagrams for CORKER are shown in Figure C3-1.

4.2 CONSTRAINTS

- (a) Derived rate had to be enabled for CORKER to function.
- (b) IRU-2 was reserved for leak clearing activities (CORKER was specifically designed for VO-1 which had both IRUs functioning).

SECTION 5

FUNCTIONAL REQUIREMENTS

5.1 INPUTS

CORKER required access to the engineering telemetry via DECOM.

5.2 PROCESSING

Position signals for all axes were monitored in the engineering telemetry stream. The ACS status word was also monitored. If any axis remained outside the limits set in CORKER (normally 2 DN) for a preset period (normally 5 minutes) the appropriate command table (Figure C3-2) was entered. The table selected depended on whether the spacecraft was in celestial or inertial mode, as determined from the status word.

5.3 OUTPUTS

Output of CORKER consisted of operational commands to the ACS and telemetry status words to the FDS. The commands were:

CC7K22	-	Derived Rate Disable
CC7K12	-	Derived Rate Enable
CC7B0120	-	IRU-2 ON
CC7B0210	-	IRU-2 OFF

Telemetry words were:

CLKW1	-	CORKER Inertial Mode Counter
CLKW2	-	CORKER Celestial Mode Counter
OPVER	-	CORKER Status

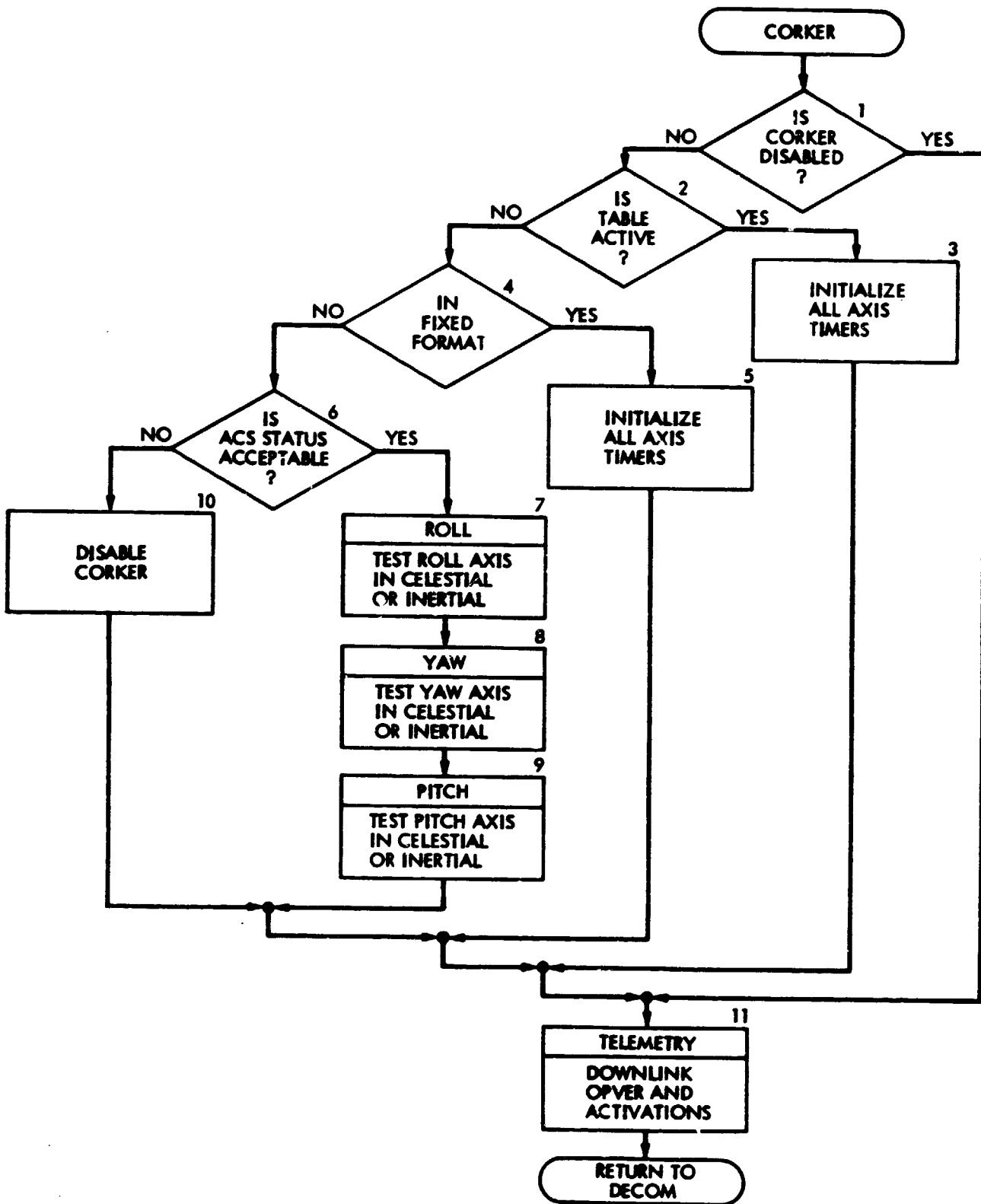


Figure C3-1. Corker Flow Diagram (Sheet 1 of 9)

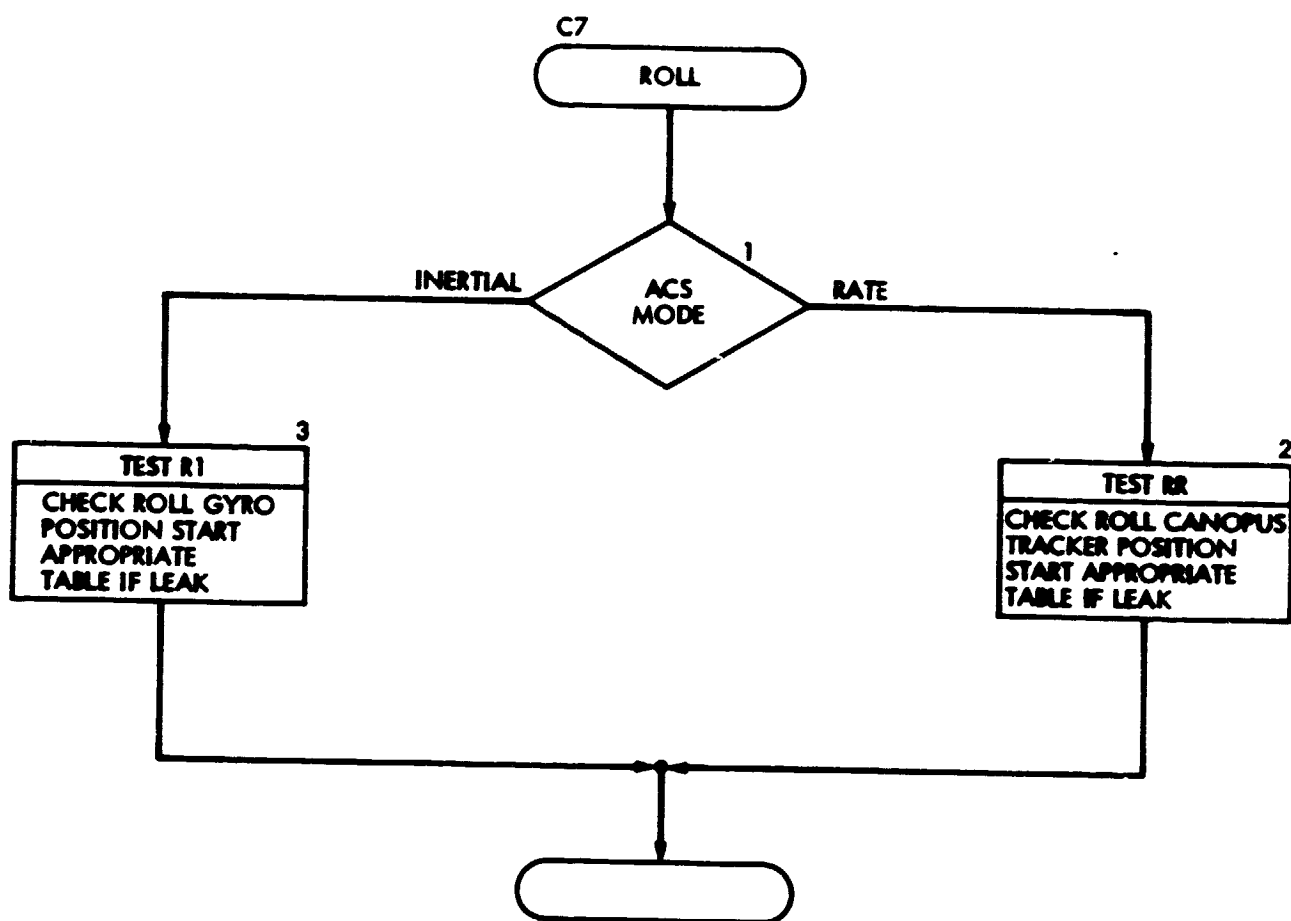


Figure C3-1. Corker Flow Diagram (Sheet 2 of 9)

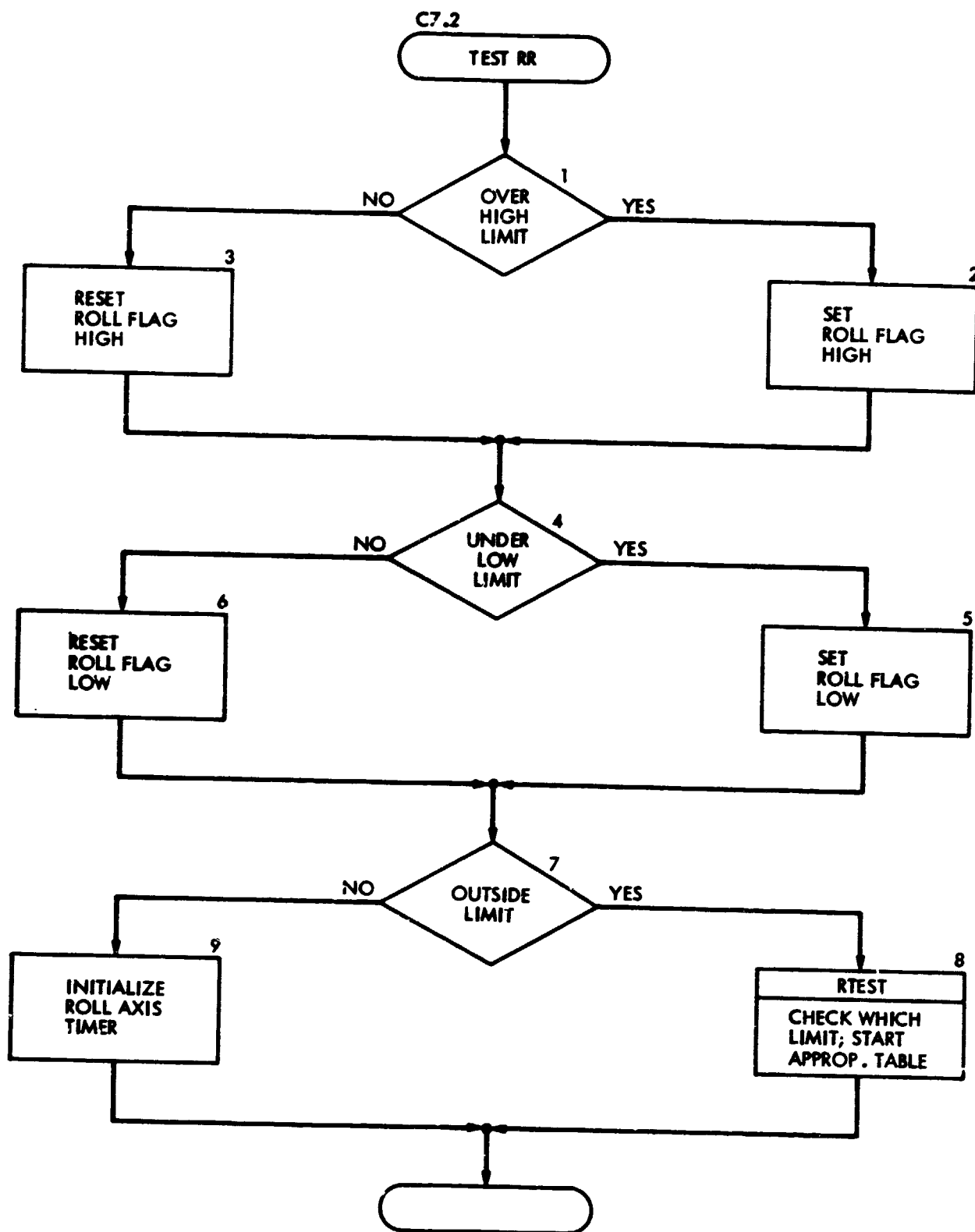


Figure C3-1. Corker Flow Diagram (Sheet 3 of 9)

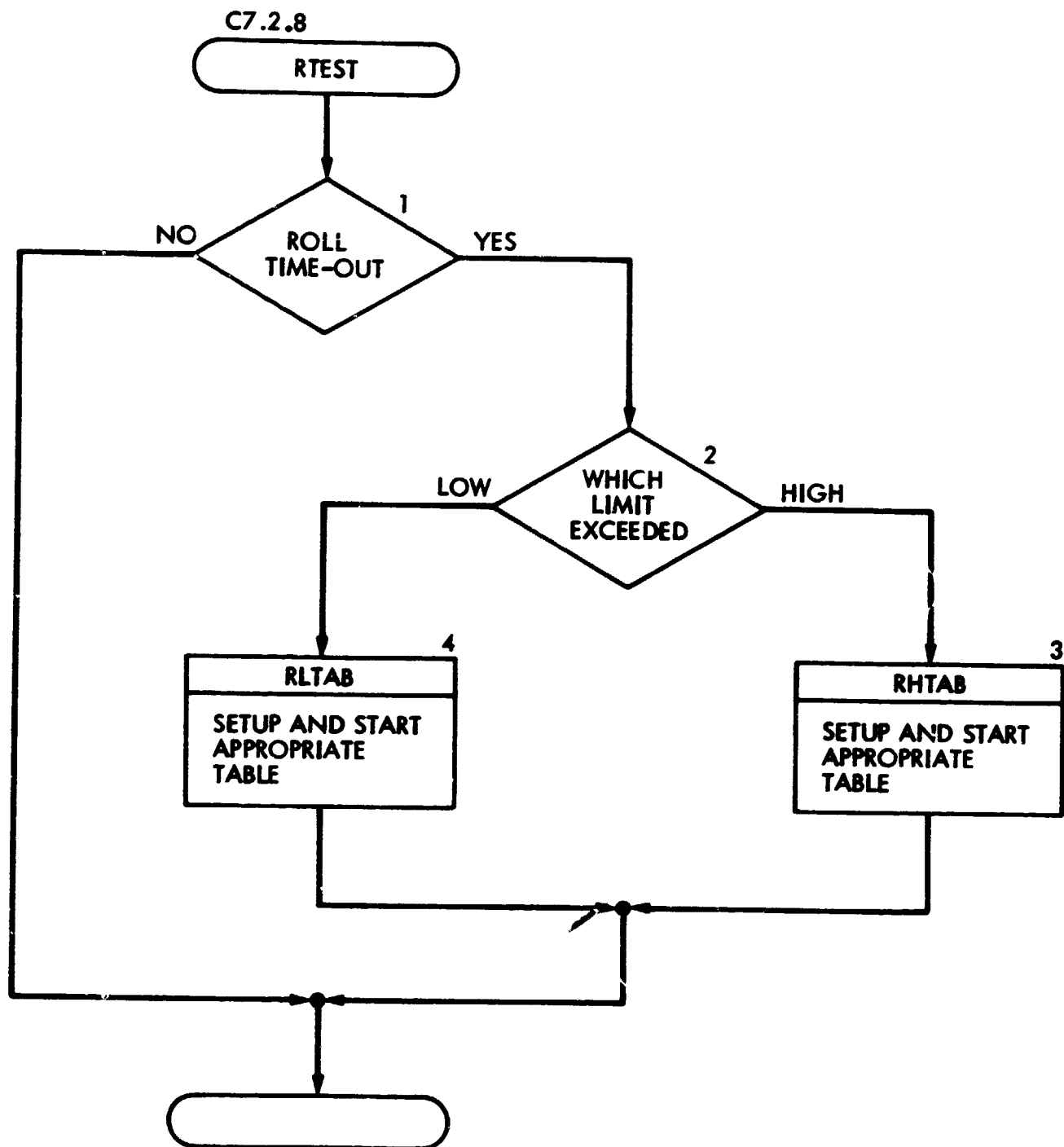


Figure C3-1. Corker Flow Diagram (Sheet 4 of 9)

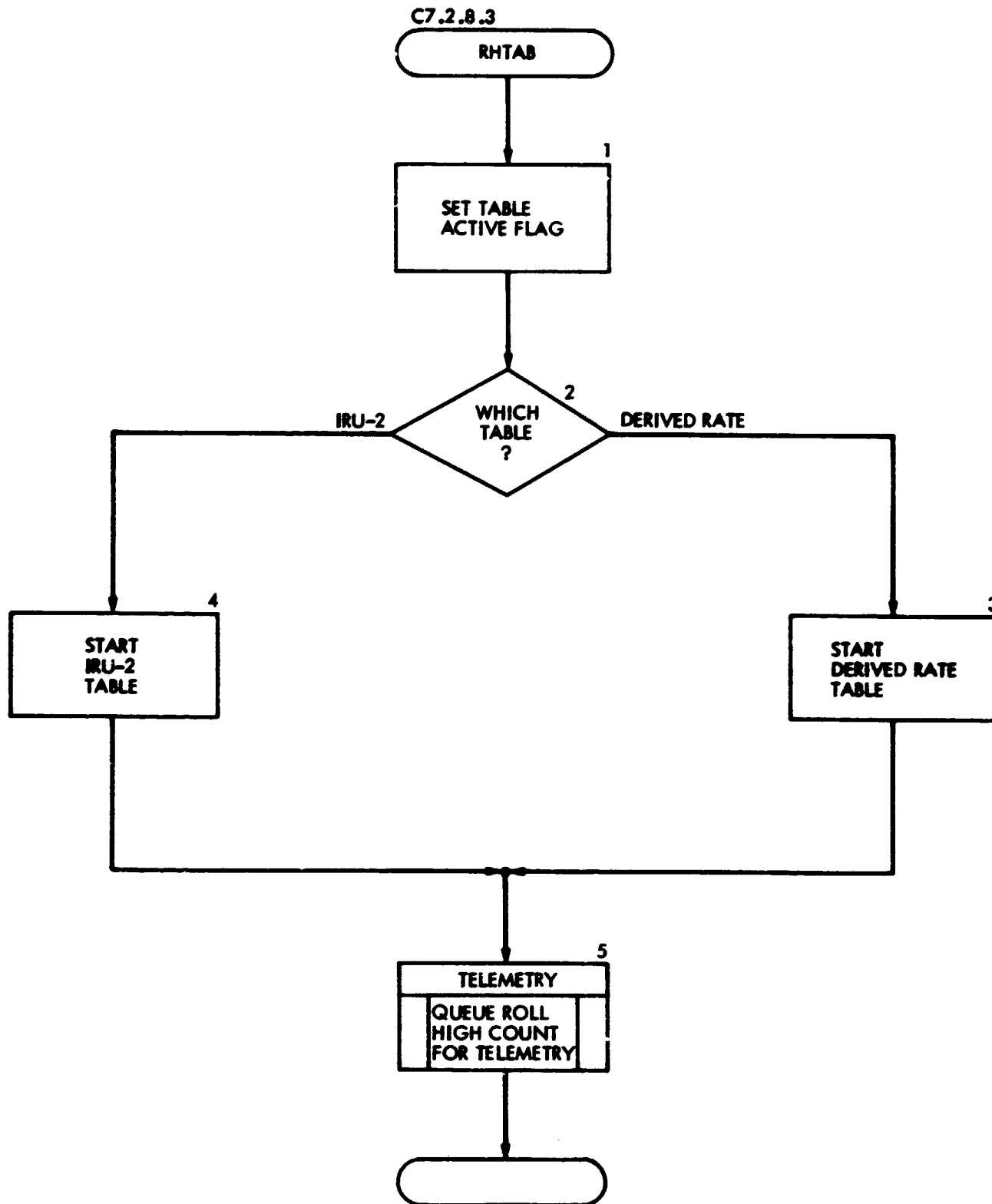
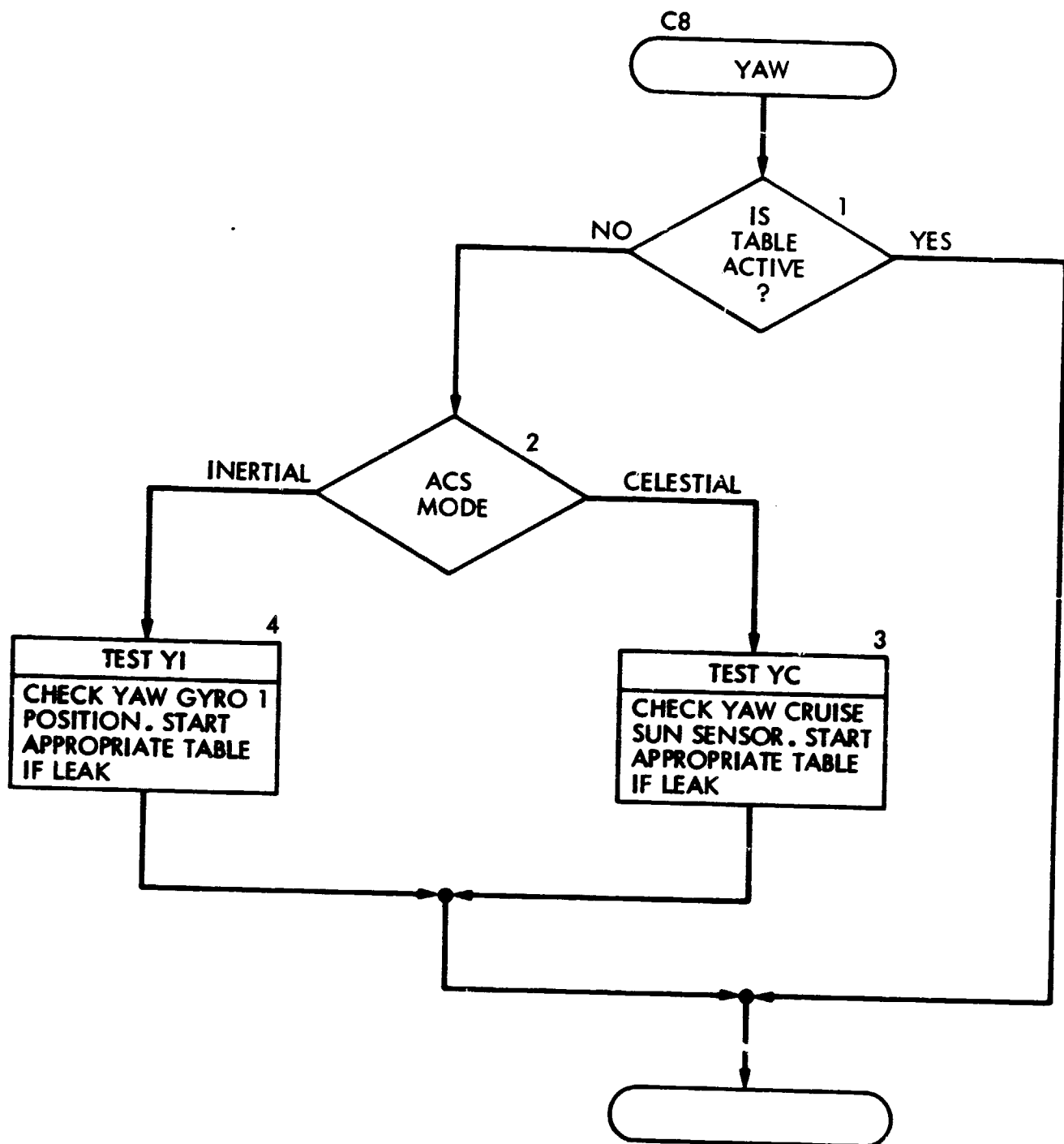


Figure C3-1. Corker Flow Diagram (Sheet 5 of 9)



NOTE: PITCH (C9) IS SIMILAR.
PITCH CRUISE SUN SENSOR
AND PITCH GYRO 1
POSITION DATA NUMBERS
ARE CHECKED.

Figure C3-1. Corker Flow Diagram (Sheet 6 of 9)

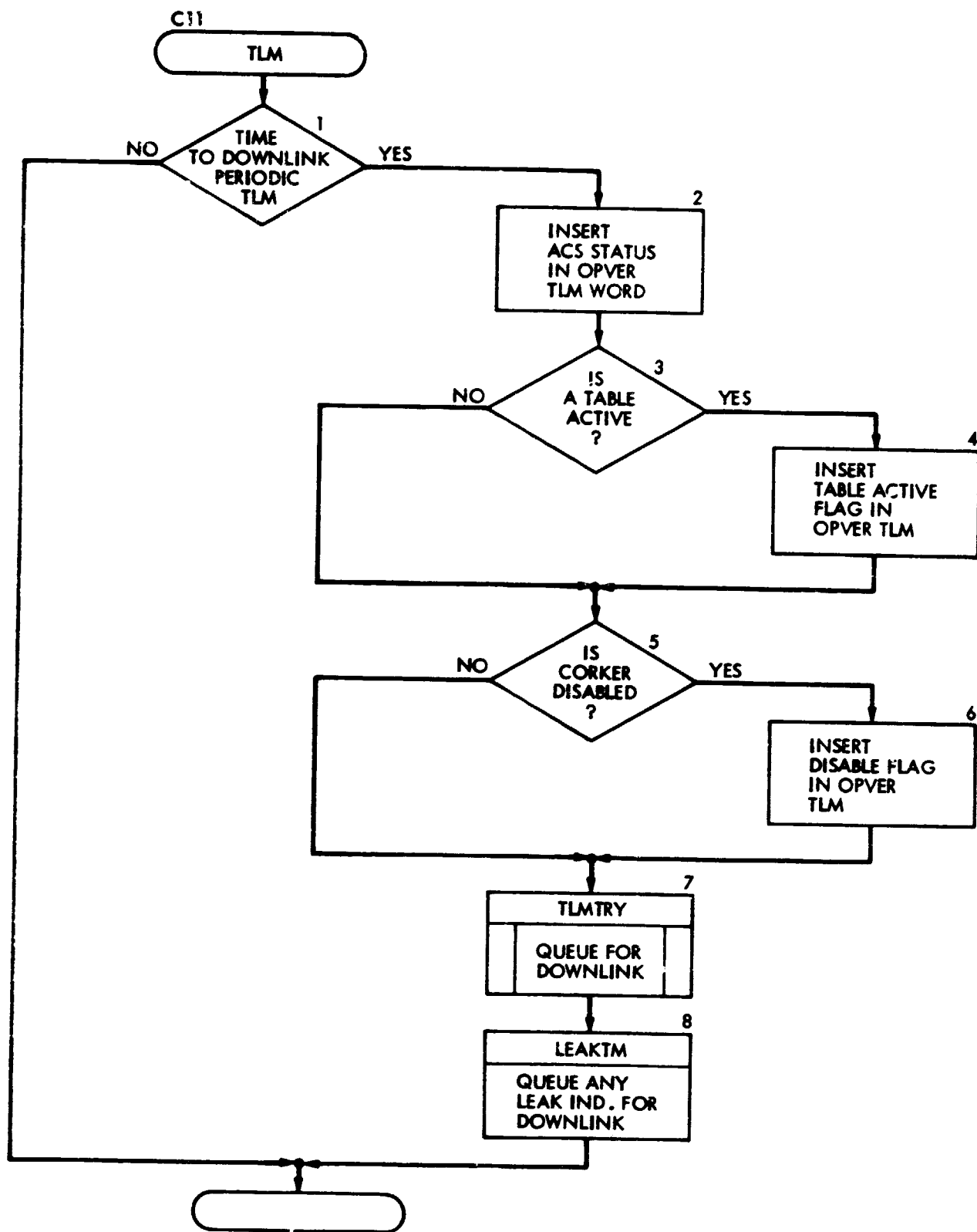


Figure C3-1. Corker Flow Diagram (Sheet 7 of 9)

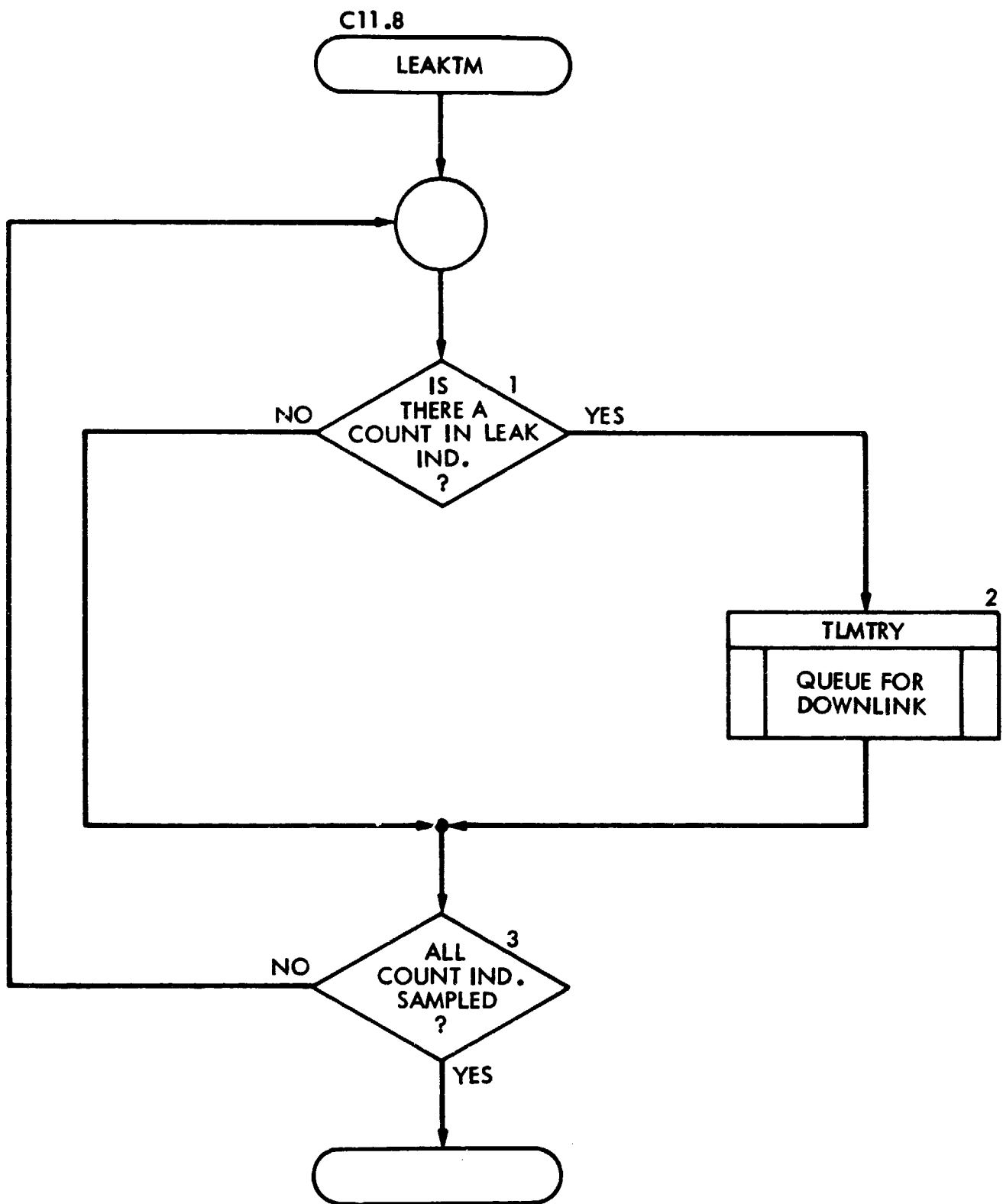


Figure C3-1. Corker Flow Diagram (Sheet 8 of 9)

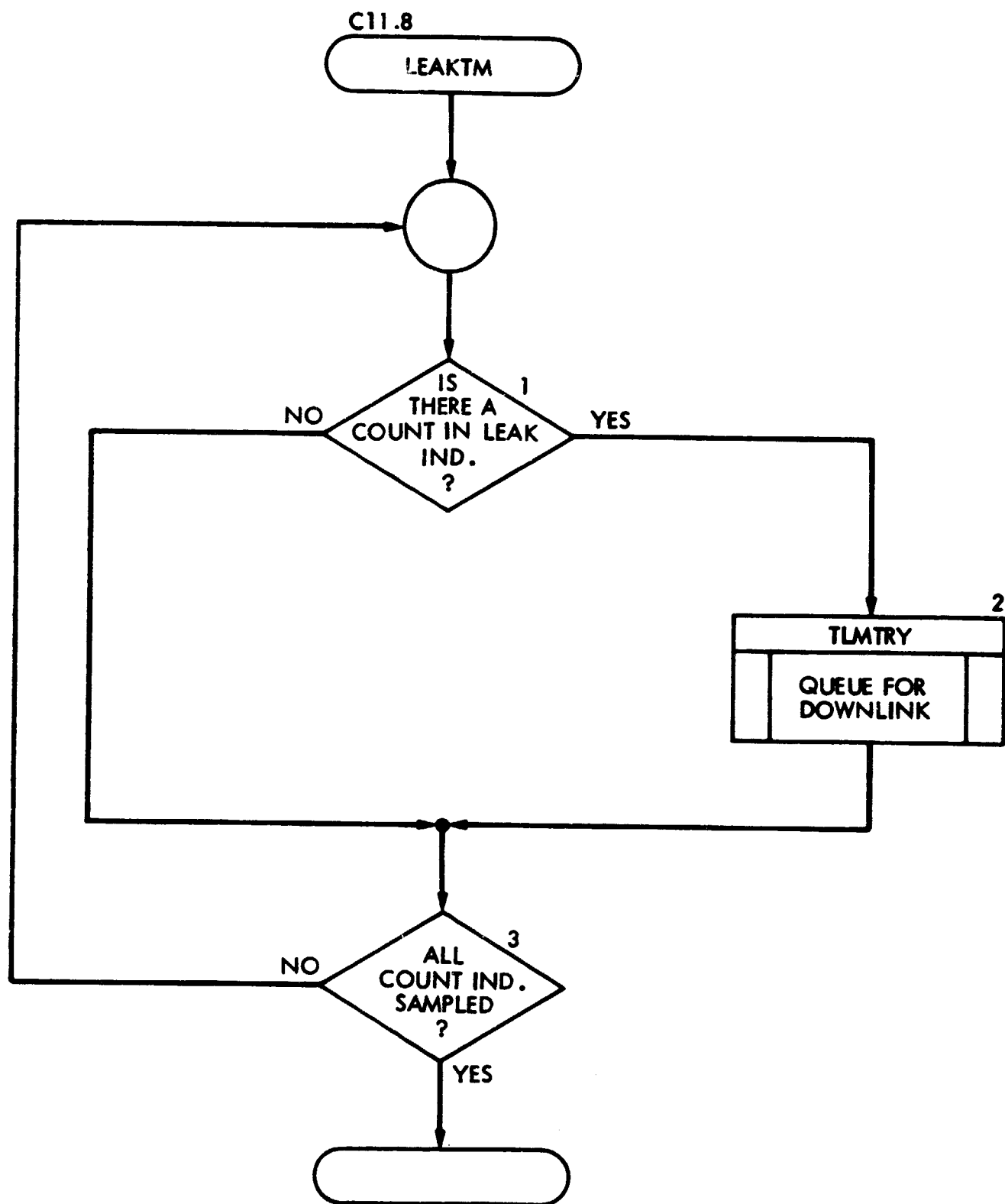


Figure C3-1. Corker Flow Diagram (Sheet 9 of 9)

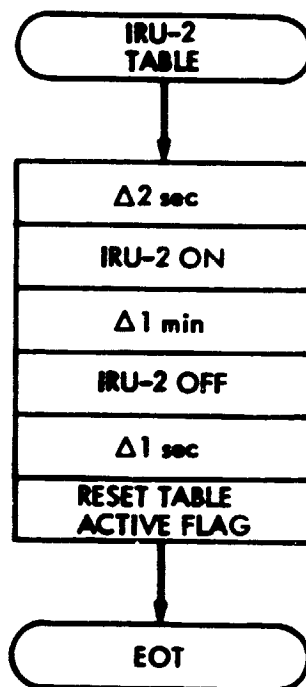
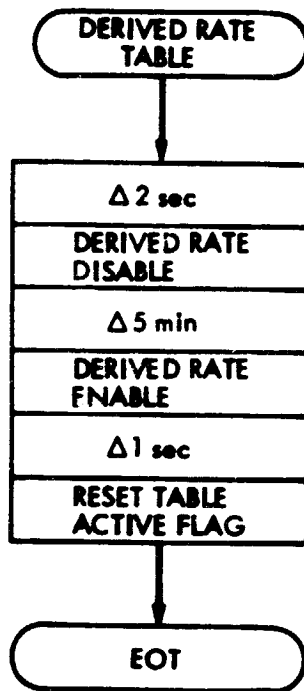


Figure C3-2. Corker Tables

SECTION 6

INTERFACES

- (a) FDS to CCS - engineering telemetry supplied to DECOM.
- (b) CCS to CCS - CORKER obtained position and status from DECOM data.
- (c) CCS to ACS - derived rate enable/inhibit and IRU-2 on/off commands.
- (d) CCS to FDS - processor telemetry containing CORKER status and activities.

SECTION 7

PERFORMANCE REQUIREMENTS

A position error of 2 DN or more outside normal deadband limits was used to indicate that a sufficiently large leak was present in the opposing jet that an attempt should be made to clear it. Selection of 2 DN was based on in-flight experience with numerous leaks, particularly on VO-2. The development of AUTO CORK, which preceded CORKER, provided an opportunity to verify the operational techniques used. The 5-minute delay before initiating leak clearing activities was also based on experience, and precluded entering a leak clearing routine during transient excursions outside normal deadband limits.

SECTION 8

IMPLEMENTATION CONSIDERATIONS

8.1 HARDWARE/SOFTWARE SYSTEM

CORKER operated in the Viking Orbiter CCS as a part of the VO-1 extended mission software. Implementation of CORKER (and previously AUTO CORK) was made possible by the development of DECOM, which accessed the engineering telemetry stream. This was the only source of spacecraft position information which could have supported development of on-board leak clearing operations during the extended mission.

8.2 RATIONALE

Implementation of CORKER was necessary to reduce attitude control gas loss due to leaking control valves. On-board leak clearing greatly reduced the work load on the small extended mission ground operations team which was also constrained to limited tracking of the spacecraft.

CORKER was a logical extension of earlier ground controlled leak clearing techniques and the VO-2 AUTO CORK routine.

8.3 FLIGHT EXPERIENCE

CORKER was active for nearly two years during the last phases of the VO-1 extended mission. Several leaks were successfully cleared. Problems were minor and were easily corrected. After CORKER was activated inadvertently during a periapsis passage due to gravity gradient torque, the control parameter for positive yaw was increased to prevent a recurrence. A change to CORKER's internal timing was also made after CORKER disabled itself during leak clearing exercises with IRU-2 on. This was not a normal spacecraft state since IRU-2 was reserved for leak clearing, and CORKER should not have tested for normal ACS state while activated. This fix was also easily made.

8.4 ALTERNATE IMPLEMENTATION APPROACHES

The techniques used in CORKER are readily adaptable to fault protection algorithms on 3-axis stabilized spacecraft using mass expulsion for control. Direct measurement of thruster leakage is difficult and has only been developed to date for very larger thrusters (e.g., Shuttle Orbiter Reaction Control System). Inferential determination of thruster leakage from spacecraft attitude control parameters, as was demonstrated using position data in CORKER, is a useful fault detection technique.

SECTION 9

VALIDATION

The CORKER software was carefully reviewed before use on VO-1. Considerable benefit was derived from earlier development of AUTO CORK. However, because of the nature of the Viking Orbiter Extended Mission, no spacecraft system test of this routine was conducted.

Appendix C
Section C4
RFL0SS Routine

DESCRIPTION OF AN AUTONOMOUS FUNCTION

SECTION 1

FUNCTION NAME: RFLOSS

SECTION 2

FUNCTIONAL DESCRIPTION

The purpose of the Radio Frequency Loss (RFLOSS) routine is to restore either S-Band or X-Band (or both) downlinks subsequent to a failure of either an exciter or transmitter. The response of this routine is prioritized; both exciters are checked before proceeding to the traveling wave tube amplifiers (TWTAs), and S-Band is always checked before X-Band. In general, the first response of this routine is to select the appropriate redundant element. The exception to this rule is the S-Band exciter; before switching to the redundant exciter, the Voltage-Controlled-Oscillator (VCO) is disabled as a source of the downlink as the input to see if that will correct the anomaly.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

The Voyager spacecraft must be maintained in a state such that it is capable of transmitting telemetry data to the ground. No single point failure mode of any component shall cause loss of all data return from more than one science instrument or the loss of more than 50% of the engineering data.

3.2 SYSTEM REQUIREMENTS

Analysis of mission requirements resulted in the need for significant block redundancy and the formulation of response priorities to direct the design. In order of decreasing priority, they are:

- (1) Spacecraft safety and commandability.
- (2) Conservation of spacecraft consumables.
- (3) Downlink telemetry visibility.
- (4) Ongoing sequence integrity.

To ensure downlink telemetry visibility, the spacecraft must exercise all combinations of downlink frequency sources, exciters and transmitters if an interrupt is sensed by the CCS after a predetermined amount of time.

SECTION 4

DESCRIPTION OF FUNCTION

Diode detectors within the Radio Frequency Subsystem (RFS) monitor the output power of the exciters and transmitters. Whenever the output power drops below a preset level (graceful degradation) the detector opens a switch.

Constraints: The turning off of a unit must not be sensed as a failed unit thereby activating the RFLOSS routine.

The routine, upon entry, must disable itself from re-entry until the downlink has been re-established.

4.1 INTERFACE DESCRIPTION

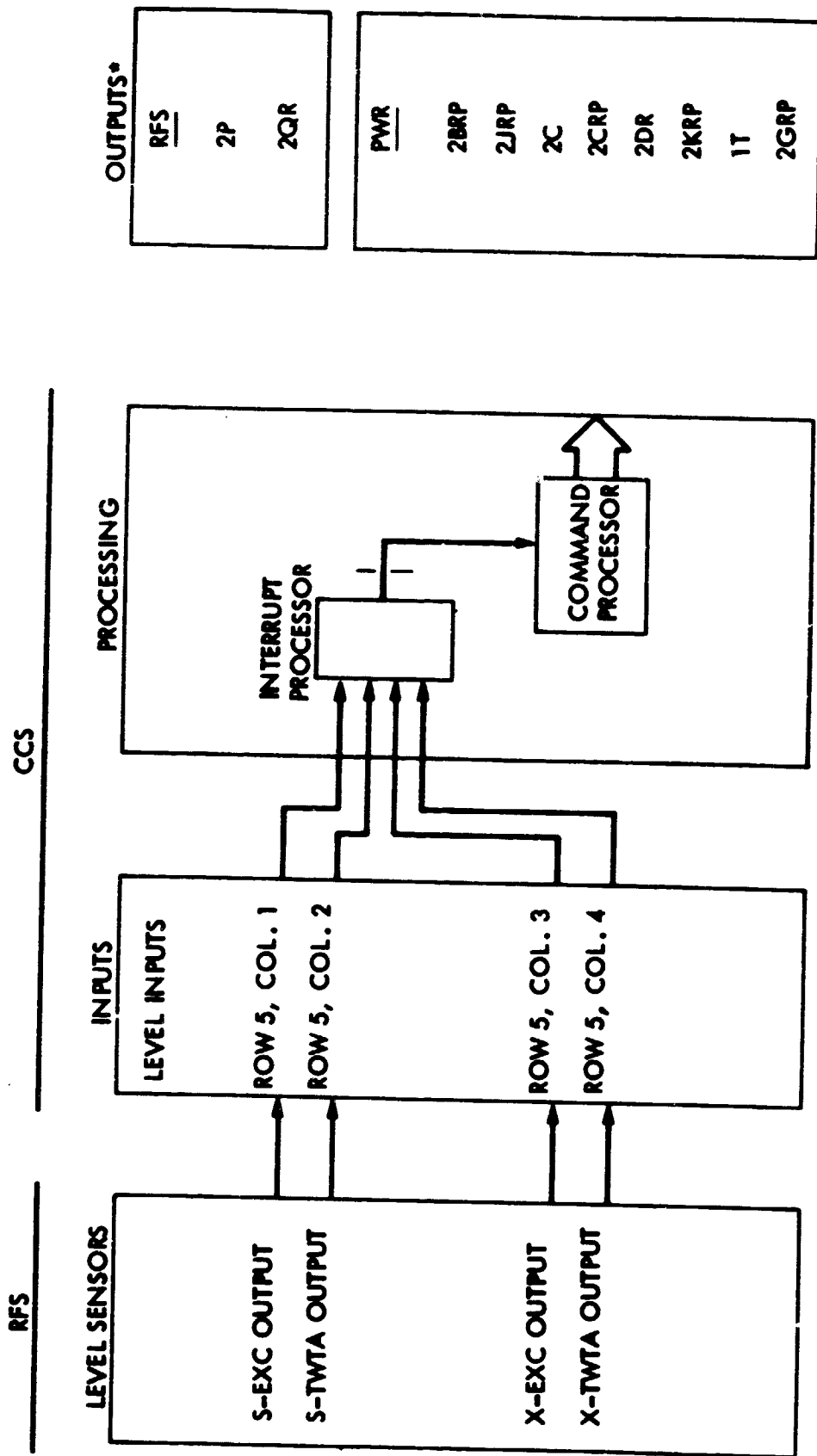
RFLOSS operates in conjunction with the Interrupt and Level Inputs of the Interrupt Processor (IRP) within the Command Computer Subsystem (CCS) to automatically reconfigure the S/C RF downlink components.

RFLOSS is entered whenever the External Level 3 interrupt indicates a change.

As shown in Figure C4-1, RFLOSS generates output commands to switch hardware within the Power Subsystem (PWR) and the Radio Frequency Subsystem (RFS).

4.2 DETAILED DESCRIPTION

The Interrupt Processor (IRP) provides the Central Processor (CP) with input information. This information is provided to the IRP as an interrupt or a level input. Interrupts are generally pulses whose occurrence is stored by the IRP until processed by the CP. The level inputs are binary levels which indicate one of the two states of each unit. Most level inputs are monitored by change detectors which cause an interrupt to occur when a change is detected.



* VOYAGER - SPECIFIC COMMANDS GIVEN FOR EXAMPLE.

Figure C4-1. RFLoss Interface Block Diagram

The IRP is continually looking through the unmasked interrupts to find the highest priority interrupt which is waiting to be processed. Where found, a number (from 0 to 31 and unique to that interrupt) is offered to the CP for processing. If accepted the interrupt is reset. The RFLOSS routine is assigned a priority level of 28.

4.2.1 Inputs

There is one interrupt processed in the RFLOSS algorithm (#28, external level 3), along with four level inputs:

<u>ROW</u>	<u>COLUMN</u>	<u>DESCRIPTION</u>
5	1	Low S-Band Exciter Power
5	2	Low S-Band TWTa Power
5	3	Low X-Band Exciter Power
5	4	Low X-Band TWTa Power

4.2.2 Processing

As shown in the flow diagrams of Figure C4-2, whenever the output power drops below a preset level the detector opens a switch. The switch opening is gated to CCS by the presence of "switched" power to the device (exciter or transmitter) such that a unit that has turned off does not look like a failed unit to CCS. Any one or more of the four interrupts will cause RFLOSS to be entered, and during the execution of the routine all four interrupts will be systematically interrogated.

Upon entry, the routine will first disable itself from re-entry, increment the master counter, and then wait five seconds before processing the exciter interrupts. (This delay permits the routine to be tolerant of exciter interrupts produced at turn-on). Following the five-second delay, the RFLOSS counter (ex post facto diagnostic trace) is incremented and the S-Band exciter interrupt is checked. If the level indicates a failure, a command is issued to decouple the exciter's input frequency reference from the ground-transmitted uplink. This will eliminate the radio's voltage-controlled oscillator as a possible failure source. One second later, the RFLOSS counter is incremented again and the S-Band exciter level rechecked. If still present, the routine will disable future entry into the S-Band exciter interrupt subroutine and issue the command to select the backup unit. The routine will wait five seconds to increment the RFLOSS counter, and check the remaining three interrupt levels. Also, one second after the exciter switch, the S-Band exciter level input is checked for the last time. If it still indicates a failed unit, the ultra-stable oscillator is turned off, thereby removing it as the last possible source of failure. At this point, the radio's auxiliary oscillator becomes the downlink frequency source.

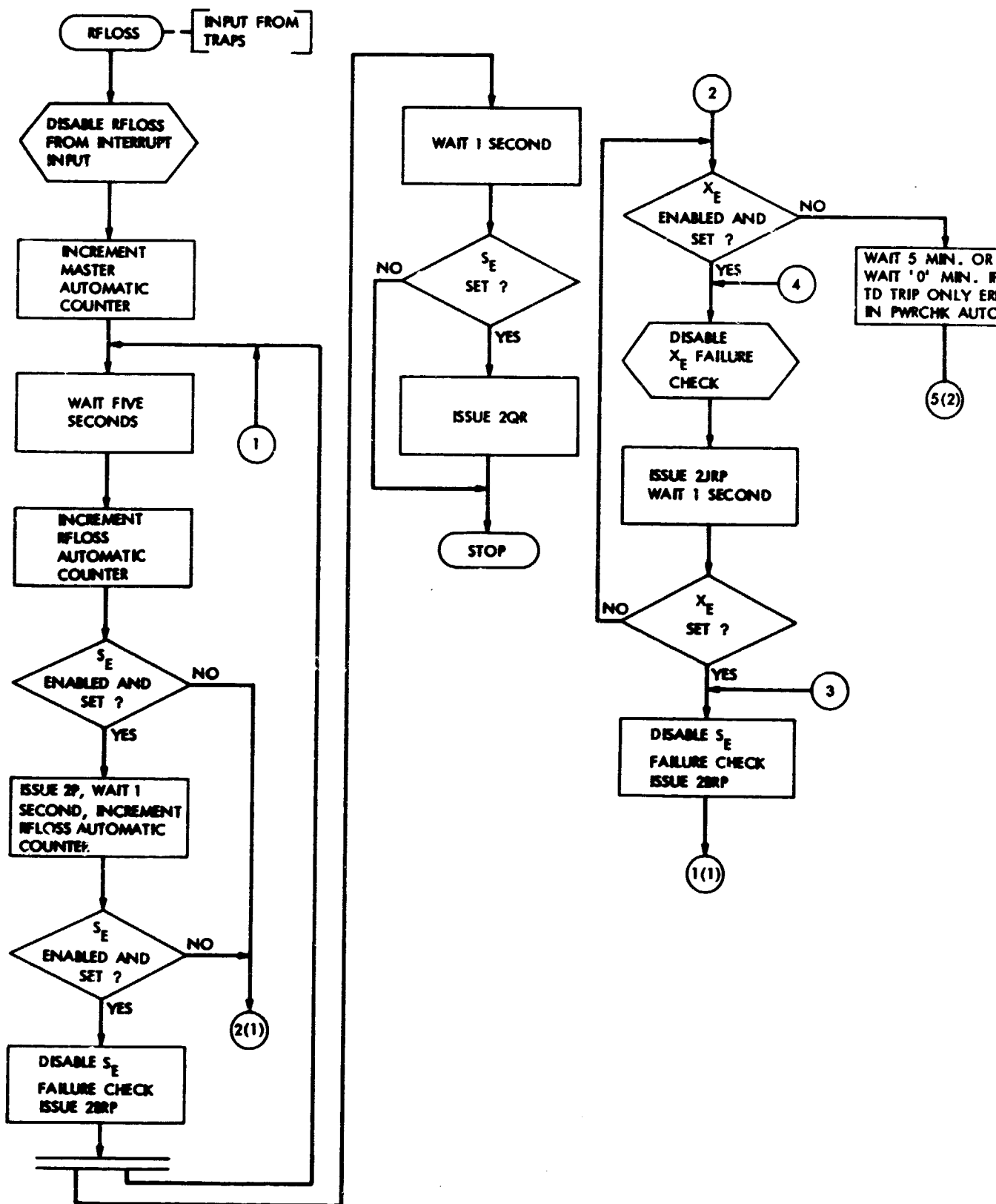


Figure C4-2. Flow Diagram, RFLOSS Routine (Sheet 1 of 2)

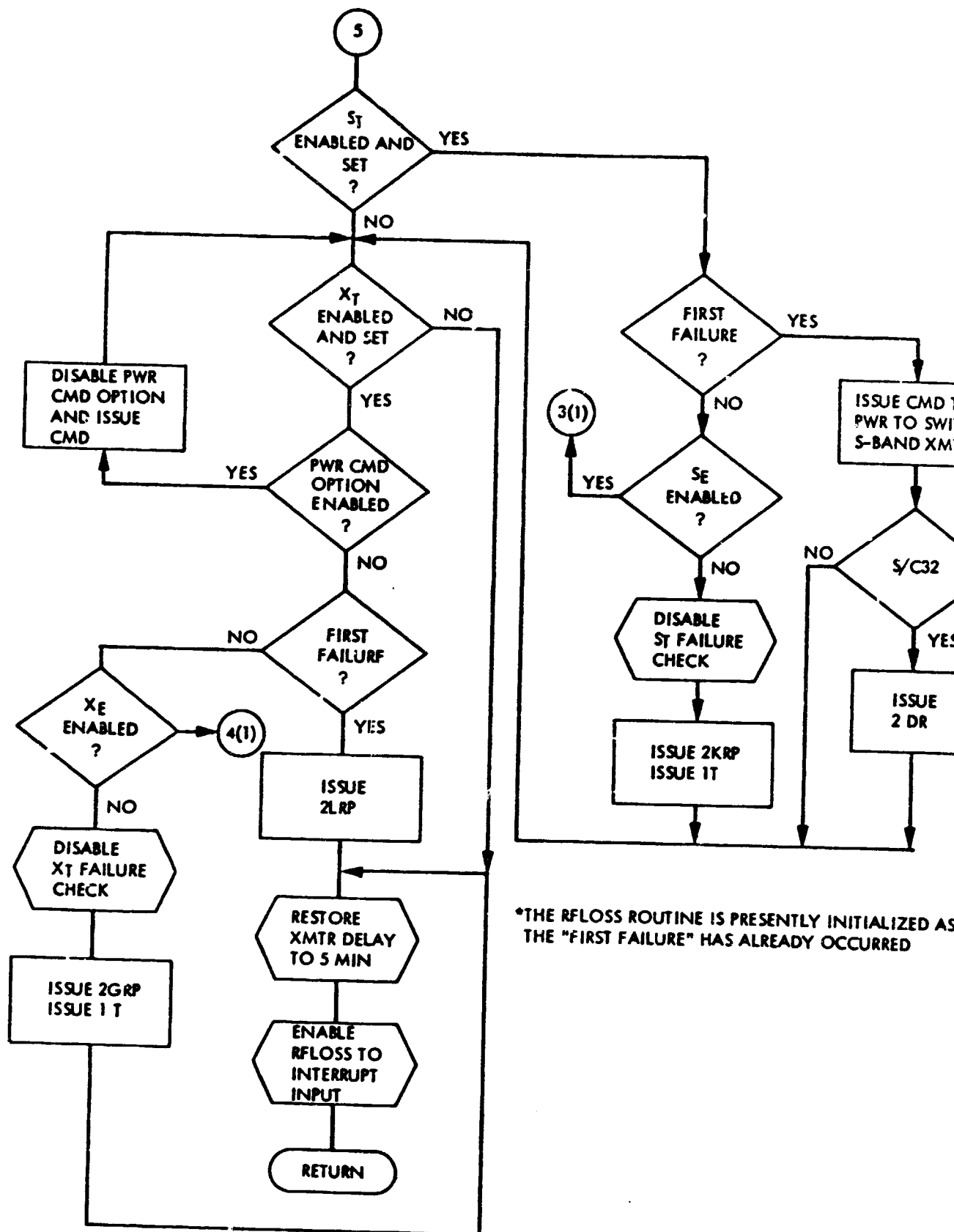


Figure C4-2. Flow Diagram, RFLOSS Routine (Sheet 2 of 2)

The next interrupt level to be processed is that of the X-Band exciter. If this interrupt indicates a failure, the routine will disable future X-Band exciter failure checks and issue the command to select the redundant X-Band exciter. If the failure indicator is still present one second later, the backup S-Band exciter (the frequency source for the X-Band exciter) is selected; future S-Band exciter checks are then disabled.

After processing the exciter level inputs, the routine moves on to check first the S-Band, then the X-Band transmitter level inputs. As with the exciters, a delay (of five minutes) is provided to assure tolerance to the transmitter's turn-on characteristics. Following the five minute delay, if the S-Band transmitter failure is indicated and it is the first indication, then the transmitters have already been switched and the suspected cause becomes the transmitter's input source, the S-Band exciter. If the S-Band exciter has not yet been switched then it will be at this time, and future exciter switches will be disabled and the routines will be re-entered back at the five second delay point (beginning). If the S-Band exciter has already been switched, then the routine will inhibit future checks of the S-Band transmitter. Following this, the routine is re-enabled and exited.

4.2.3 Outputs

VOYAGER PARAMETER MNEMONIC	PARAMETER DESCRIPTION	TYPE
2P	Two way non-Coherent on	Discrete
2QR	Ultra-Stable Oscillator off	Discrete
2BRP	S-Band exc 2 select	Coded
2JRP	X-Band exc 2 select	Coded
2CP	S-Band TWTA 1 select	Coded
2DR	S-Band TWT Low Power	Coded
2KRP	S-Band TWTA off	Coded
IT	Bag 1 heater on	Coded
2GRP	X-Band XMTR off	Coded

4.3 IMPLEMENTATION CONSIDERATIONS

4.3.1 General Description of the Hardware/Software System

RFLOSS is a software routine executed by the spacecraft Command Computer Subsystem (CCS). As used on Voyager, the CCS serving as a central controller is composed of two computers, each of which is used as an interrupt processor, reacting to periodic timing interrupts (hours, seconds, centiseconds, science data frame timing, command bit sync, etc.), and external level interrupts from other subsystems which are typically used to indicate external failures elsewhere in the spacecraft. Both processors have an 18-bit, plated-wire (hence nonvolatile) memory containing 4096 words, half of which are "write protected" such that a "key" must be employed any time this part of the memory is to be altered. Fixed routine for downlink telemetry loss correction (RFLOSS) is typical of the function located in write-protected memory. The remaining half of the memory is used to load sequences which control the spacecraft's engineering and science subsystems during trajectory correction maneuvers, science data acquisition and transmittal, and various calibration exercises. Key RFLOSS interfaces with CCS are shown in the block diagram in Figure C4-1.

RFLOSS routine memory storage requirements are 93 words. Experience to date is that the routine has been entered numerous times, but has reset after the built-in 5 minute delay for a TWTa low power interrupt. The TWTa low power interrupt is set each time the TWTa power mode is switched (low to high, high to low).

4.3.2 Rationale for Technique Chosen

Due to the long round-trip light-time at encounter (>2 hours for Saturn) plus analysis time, and corrective action command generation time, an autonomous recovery function was required. Diode detectors very similar to the Telemetry (TLM) channels for S-Band and X-Band power output were used to detect changes in power output. Also, similar circuits were used for S/C exciter drive power output.

An additional reason for the function was the possibility of the hydrazine line freezing if power were lost from the TWTa's or if the power output persisted in the low power state. The interrupts were set so that if the TWTa's degraded gracefully the downlink TLM of 40 bits per second (bps) uncoded on S-Band could be supported by the 63-meter Deep Space Network (DSN) antenna thru Saturn Encounter.

4.3.3 Flight Modification

There have been no modifications made to the RFLOSS routine since launch.

4.3.4 Impact of New Technology/Alternate Implementation Approaches

One thing missing from this routine is an autonomous correction for loss of the S-Band and/or X-Band downlink modulator, or Telemetry Modulation Unit (TMU) failure. The TMU redundant telemetry drives are both presently active on the Voyager S/C.

SECTION 5

VALIDATION AND TEST

The test matrix is shown in Table C4-1.

5.1 ANALYSIS REQUIREMENTS

The setting of the interrupt level had to be selected such that a graceful degradation could be maintained if only one unit remained, or for some other reason a degraded performance was acceptable to using a redundant unit. The value chosen was such that 40 bps could be supported using a 64-meter Deep Space Station (DSS) through Saturn Encounter.

5.2 TEST REQUIREMENTS

5.2.1 Unit/Subsystems Test

The test demonstrated that:

- (1) The RFLOSS routine will operate with a worst case mission load functioning in the same time frame, in the same computer.
- (2) Mission sequence, if inhibited for S/C safety, will resume automatically. The RFLOSS will not inhibit an on-going sequence.
- (3) A memory check sum of the RFLOSS routine is not performed every sixty (60) days even though this would verify proper operation.

5.2.2 System Test Sequences

These test sequences were executed to demonstrate that:

- (1) Power and RFS switching will not inhibit RFLOSS routine operations.

Table C4-1. RFLOSS Test Matrix

REQUIREMENTS	UNIT/SUB-SYSTEM TESTS	INTER-FACE TESTS	SYSTEM TEST #1	ENVIRONMENTAL TESTS	SYSTEM TEST #2	MISSION TESTS	ETR SYSTEM TEST	LAUNCH COUNT DOWN	REMARKS
INITIAL RFLOSS CONFIGURATION	X			*X					* PERFORMANCE AT VENDORS FACILITY INTEGRATION CHECKS.
RFLOSS RECOVERY	X		X		X				
GRID ENABLE/DIS-ABLE						X			
TEMPERATURE CONTROL			X		X				
RF'S SELECTION	X		X		X				REDUNDANT STRINGS
POWER SELECTION		X		X					REDUNDANT POWER RELAYS

- (2) Electromagnetic Interference (EMI) and static discharge test environments did not inhibit RFLOSS operations.
- (3) RFS power overloads will not inhibit RFLOSS operations, although these were not tested.
- (4) The RFLOSS routine will be automatically inhibited during memory refresh periods. However, this was also not tested.
- (5) Only a single RFLOSS routine is active at any time.
- (6) Switching off an element of the RFLOSS routine does not cause the routine to switch to the redundant element.

5.2.3 External Interface Tests

The external interface tests with the CCS were not developed due to hardware scheduling and costs. This test was scheduled to be performed in the Telecommunications Development Laboratory (TDL) where the hardware/software interface would be exercised. Instead a hardware interface wiring and switching test was performed at subsystem integration to the S/C bus.

An additional test sequence was performed at the vendor's facility to verify proper level setting of the interrupt over-temperature during a thermal-vacuum test.

5.2.4 Mission or User Level Tests

The mission test sequence demonstrated downlink telemetry loss recovery from two (2) different CCS and RFS configurations.

5.3 SIMULATION REQUIREMENTS

Ground support special test equipment was used to simulate a downlink telemetry loss to activate the RFLOSS recovery routine.

5.4 VALIDATION IMPLEMENTATION REQUIREMENTS

5.4.1 Ground Support Equipment

To verify proper RFLOSS operation, a special test box and cables were used to simulate a loss of downlink carrier by overriding the diode voltage on each level interrupt.

5.4.2 Test Facilities: N/A

5.4.3 Special Test Hardware and Software

The special test hardware consisted of adjustable d.c. power supplies and switching so that each diode detector level interrupt could be activated.

5.4.4 Test Operations Requirements

- (1) Launch Vehicle Safety Requirements: The RFLOSS routine was inhibited until after vehicle separation due to critical pressure potential failure of the S-Band TWTAs.
- (2) Fail Safe Requirements: The switching off of an element will not cause the RFLOSS routine to issue commands to the redundant units.
- (3) Command Errors: No single ground command can inhibit the operation of the RFLOSS routine.
- (4) Self Test: N/A.

Appendix C
Section C5
Celestial Sensor Fault Detection/Protection

DESCRIPTION OF AN AUTONOMOUS FUNCTION

SECTION 1

FUNCTION NAME: CELESTIAL SENSOR FAULT DETECTION/PROTECTION

SECTION 2

FUNCTIONAL DESCRIPTION

The Celestial Sensor Logic monitors the operations of the Sun Sensor (SS) and Canopus Star Tracker (CST) now in use on Voyager, and, upon detection of error, triggers the reacquisition of celestial references and triggers changes in the Attitude and Articulation Control Subsystem (AACS) configuration. Configuration changes include the swapping of Hybrid Buffer Interface Circuits (HYBICs) and their dedicated sensors.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

The spacecraft must be able to maintain its attitude during all phases of the mission. During interplanetary cruise, celestial sensors, the SS and CST, are used to provide attitude information. The health, welfare and proper operation of these sensors shall be maintained.

3.2 SPACECRAFT REQUIREMENTS

The SS locks the spacecraft (S/C) to the Sun and provides pitch and yaw control. The CST locks the S/C to Canopus, and provides roll control. To have the SS and CST provide attitude control sensing capability, the AACS shall maintain the integrity of SS and CST by detecting errors, initiating reacquisition of celestial references when required, and swapping to redundant AACS components. (Current logic does not provide for CST redundant sensor swap.) The AACS shall notify the Computer Command Subsystem (CCS) through the power code interface, upon the loss of and upon reacquisition of the Sun and/or Canopus. Commanded turns to perform initial acquisition or reacquisition Sun searches will be stored in CCS. CCS shall initiate all searches.

3.2.1 Sun Sensor

The sun sensor (SS) is packaged as a single unit to provide a two axis, electrically biasable, attitude position sensing capability. The SS provides redundant pitch and yaw axis sun intensity signals, and attitude

position error signals to Hybrid Programmable Attitude Control Electronics (HYPACE) for S/C attitude control. An "IDET acquisition" state is provided by HYPACE to CCS via a power code when the sun is within the field-of-view of the SS illumination detectors (IDETS) for both pitch and yaw axes. When this occurs, the angle detector (ADET) generated position error signal is used by HYPACE for position control. An "ADET acquisition" state provides HYPACE to CCS via power code when the sun comes within the earth biased limit cycle deadband in both axes. The ADETs on both axes of the SS can be biased as required by HYPACE in order to point the high gain antenna at Earth.

Detectors: The SS contains two identical detector packages, one for the pitch and one for the yaw axis. Each detector package contains four detector elements as shown schematically in Figure C5-1.

Signal Processing Electronics: An electrical functional block diagram for a single non-redundant axis of the SS is given in Figure C5-2. The SS consists of four such identical circuits, one each for pitch and yaw dedicated to HYBIC 1, and one each for pitch and yaw dedicated to HYBIC 2, with the exception that there is only one detector excitation modulator and only one IDET circuit for both circuits associated with a given HYBIC.

The IDET provides an approximate zero voltage signal when the sun is outside the IDET field-of-view (FOV). When the sun is within the IDET FOV, the IDET provides a positive voltage sun intensity signal whose magnitude will be a function of illumination level determined by the S/C distance from the sun. The HYPACE establishes appropriate sun intensity voltage gate levels in order to assure unambiguous transfer of pitch and yaw attitude control to the SS ADET circuits at the appropriate time.

Optomechanical: The SS optomechanical design consist of SS and detector package windows to define the optical bandpass region and protect the detectors, field stops, diaphragms, and masks to define the IDET and ADET FOV's; baffles and coating to suppress stray light; slit aperture assemblies to admit and shape incident sunlight into narrow slit images for imaging onto the detectors; and detector elements to sense the sun in a narrow spectral region.

3.2.2 Canopus Star Tracker

The CST assembly consists of redundant trackers to provide roll axis error signals, cone angle position signals, and star intensity signals to HYPACE. The roll axis control signal allows the S/C to maintain roll axis alignment during celestial cruise, while the star intensity signal provides identification of stars within the scanned FOV in terms of detected brightness. Intensity gates are set in the HYPACE software to provide for star identification. Cone angle position signals identify in which of the five discrete cone positions the scanned FOV is located. The Canopus star tracker (including the sun detector/shutter) receives its power from the power subsystem (PWR).

A functional block diagram of the CST is given in Figure C5-3.

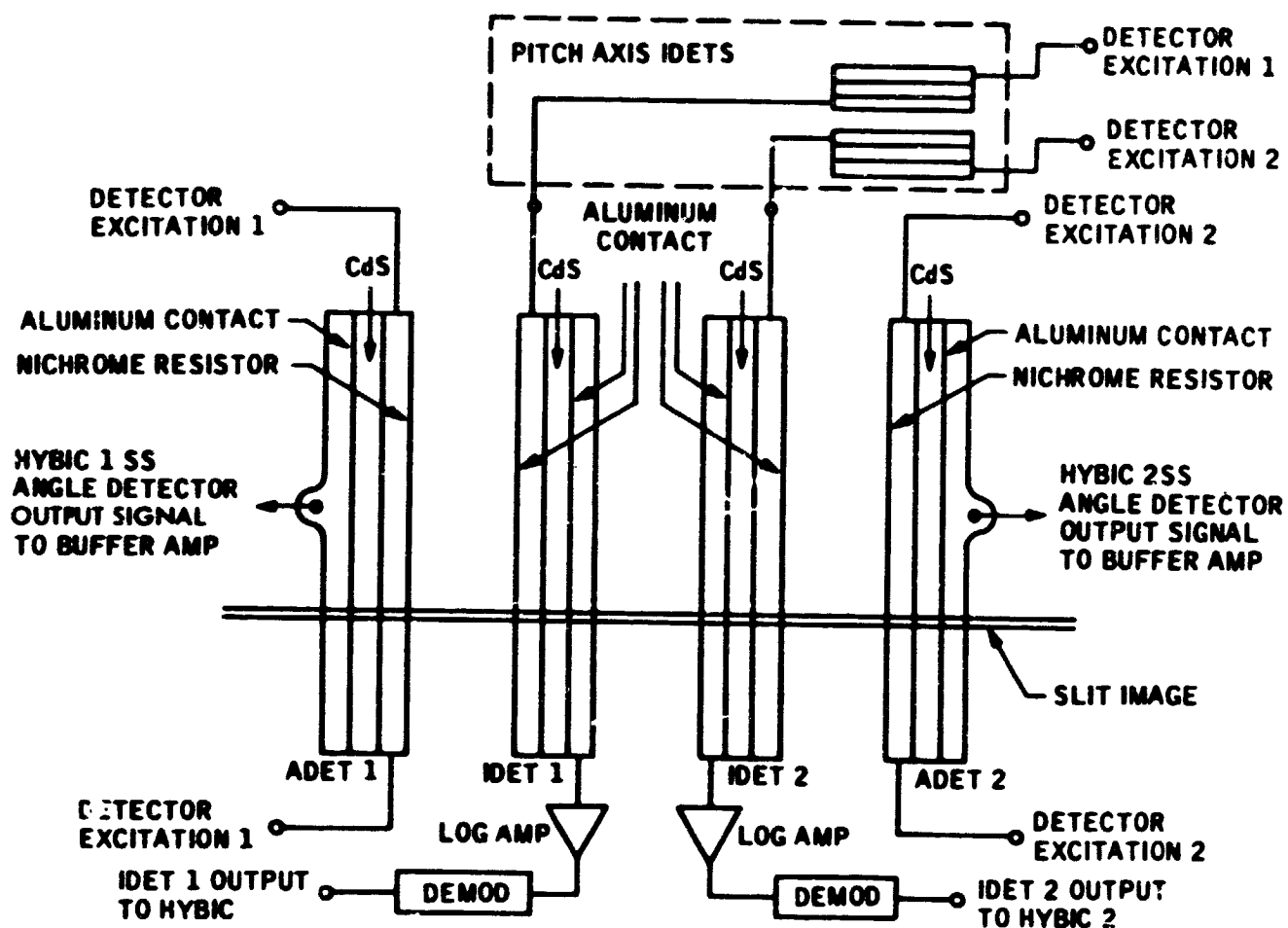


Figure C5-1. ADET and IDET Arrangement Within the Detector Package (Single Axis Redundant)

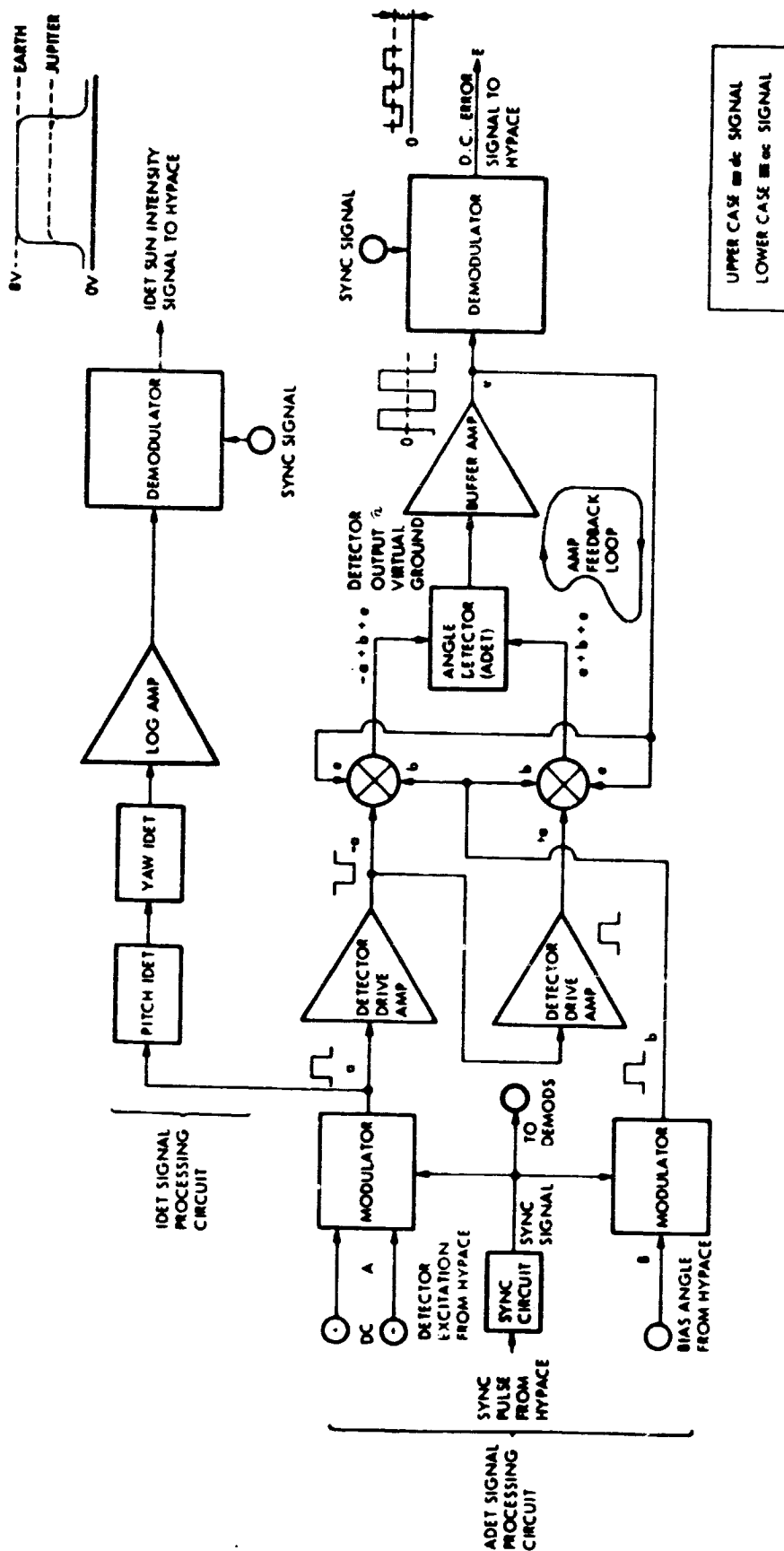


Figure C5-2. Sun Sensor Functional Block Diagram, Single Axis Nonredundant

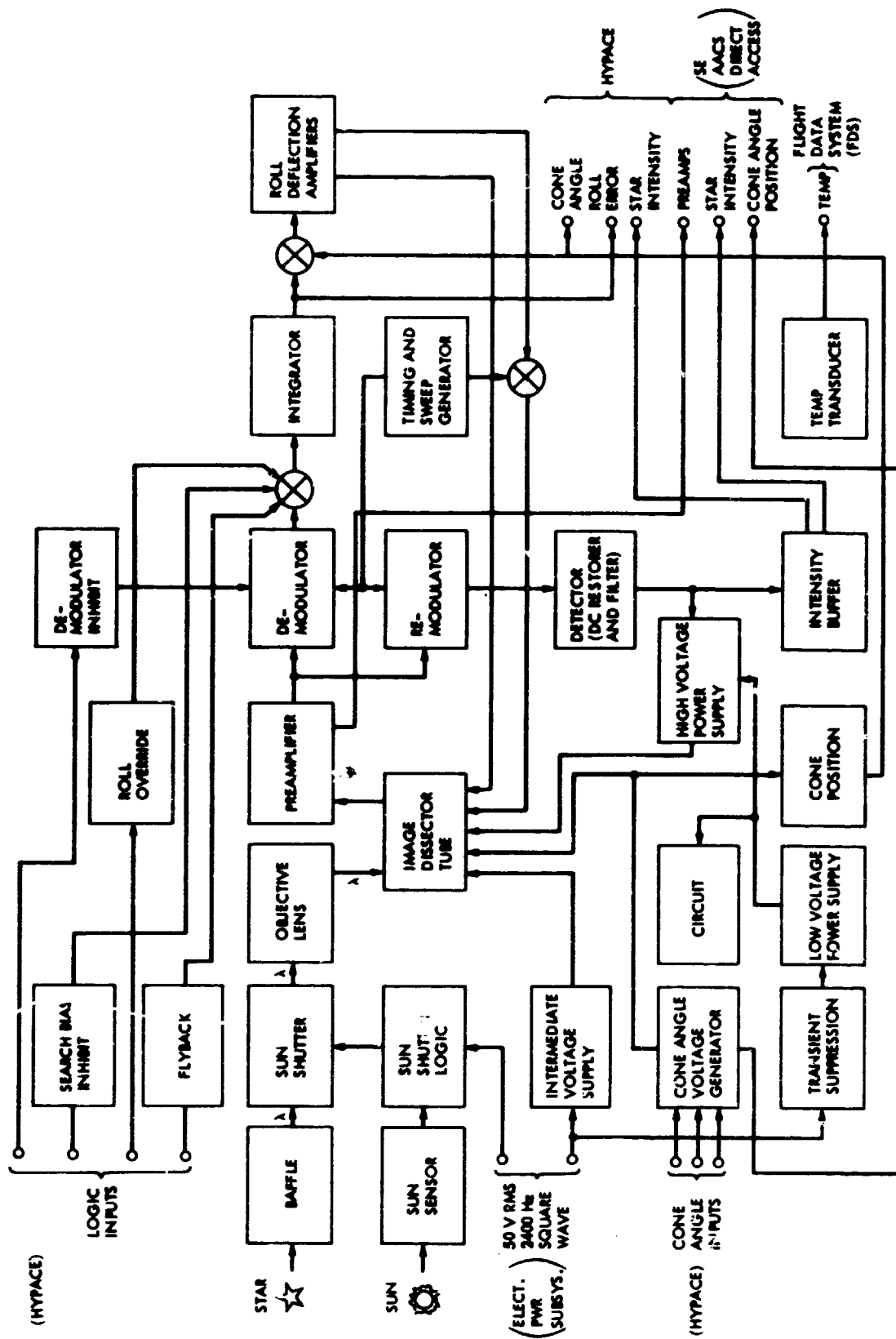


Figure C5-3. Canopus Star Tracker Functional Block Diagram

SECTION 4

SUBSYSTEM FUNCTIONAL OPERATION

Figure C5-4 shows the mode sequencing for automatic reacquisition of sun and/or Canopus. ADET loss (ADET) implies that the sun is outside the sun sensor linear FOV or the sun sensor has failed. IDET loss (IDET) implies that the sun is outside the illumination detector FOV, or the sun sensor has failed. Canopus Acquisition (CA) loss implies the star is outside the CST FOV, or the tracker has failed. IDET and ADET loss in the roll inertial mode (RI) or celestial cruise mode (CC) causes the mode to be switched to all axis inertial mode (AAI). CA loss in celestial cruise causes the mode to be switched to AAI. The six power codes (ADET, ADET, IDET, IDET, CA, AND CA) are only sent when the state changes. There are three loss routines stored in CCS. The Canopus loss routine basically, after preconditioning the Mission Module (MM), commands a roll search. After preconditioning the MM, the ADET loss routine basically waits a fixed time for ground intervention, then switches HYBIC and sun sensors. After preconditioning the MM, the IDET loss routine switches HYBIC and sun sensors, then issues a series of stored commanded turns which result in 4 π steradian coverage, thus causing IDET to again be satisfied. The three routines in CCS are prioritized, with IDET loss being the highest priority and CA loss being the lowest priority. Thus if two of the "loss" power codes exist in CCS at the same time, only the routine with the highest priority will be executed.

4.1 SUN SEARCH LOGIC

Figure C5-5 shows a high-level flow chart of the Celestial Sensor Logic. Provided that the AACS control mode is not Launch or Propulsion Mode (PM), the logic will continually determine (based on SS outputs obtained by the SS read routine) whether the IDET (Intensity DETector) signal indicates the presence or absence of the Sun within a ± 25 deg. field of view, and whether the ADET (Angle DETector) signals are within ± 2 deg. of the desired null in both pitch and yaw. The ± 3 deg. linear ADET field of view is biasable in a range of ± 20 deg. Both the IDET and ADET outputs of the SS are "filtered" through sets of acquisition-delay timers so that momentary "glitches", such as those occurring during power turn-on, and other noise, do not cause high-frequency chattering of the acquisition signals. The "ADET acquired" condition requires the ADET error in pitch and yaw to be less than 2 deg. for 30 s, whereas the loss of ADET will be signaled after one or the other errors have exceeded 2 deg for 5 s. IDET "acquire" and "loss" delay times are 2 s each.

Unless a direct command (Sun Search) is issued by CCS that SS error signals are to be used for pitch/yaw control, the remaining acquisition logic is bypassed. In the "search" situation, two possibilities are provided for. AACS may already be using SS error signals for pitch/yaw control, i.e., it is in the Roll Inertial or Celestial Cruise mode. Or, AACS may still be using gyro control in all axes for Trajectory Correction Maneuvers (TCM) or AAI modes, and in the process of searching for the Sun, or at least waiting for the delayed "IDET acquired" signal.



In the latter case, after the Sun is finally detected, any "search" turn that may have been in progress to locate the Sun is immediately stopped, and a change to the Roll Inertial control mode is executed. This switches pitch/yaw SS error signals into the Control Law and allows the ADET signals (even though they may be saturated) to drive the craft toward the null position. A power code is also sent to CCS signalling "IDET acquired". On the other hand, the absence of Sun intensity will trigger an "IDET loss" power code preceded by the "Omen." The loss condition will also cause CCS to issue a preprogrammed series of turn commands to AACS designed to search the entire celestial sphere for the "lost" Sun.

Assuming Roll Inertial or Cruise mode has been achieved, the IDET signal is still continually checked for proper level, and the "ADET acquired" signal is also now examined. It is in this portion of the logic that the loss of either ADET or IDET acquisition can eventually result in the swapping of HYBICs and their associated sensors.

For example, once Roll Inertial or Cruise mode has been established, a loss of Sun intensity will trigger an immediate HYBIC swap, and, therefore, substitution of the redundant Sun sensor. Since the IDET output must normally fall between preset high and low "gate" levels, the common circuit failures (i.e., a saturated output or zero output) should be easily detected. If the HYBIC swap and new sensor are not able to clear the IDET problem in 24 s, the AACS control mode is changed to all-axis inertia' in a last-resort effort to ensure stable attitude control of the vehicle while ground controllers diagnose the problem.

The loss of ADET, i.e., pitch or yaw angular errors greater than 2 deg for 5 s, will, in addition to causing "Omen" and "ADET loss" power codes to be issued, immediately change the control mode to Roll Inertial (if it is not already there). Going to Roll Inertial eliminates the need to contend with any problems that ADET loss may have induced in the star acquisition logic, and assures full thruster torque (nonpower-share condition) for any necessary reacquisition maneuvers. Also, it adheres to an operational ground rule that sun acquisition and star acquisition should always be performed in exactly that order before allowing the Cruise control mode to be re-established. Upon loss of ADET, a 5-min timer is started, allowing sufficient time to reacquire under the assumption that something other than a sensor failure, such as Trajectory Correction and Attitude Propulsion Unit (TCAPU) thruster failure, is the cause. If, at the end of 5 min, no improvement has been detected, a HYBIC swap will be ordered to bring in the redundant SS. Again, as in the case of IDET loss, if the swap does not successfully clear the fault after another 5 minute period, the all-axis inertial mode is ordered as a last resort.

Clearly, if the TCAPU fault correction control routine is enabled, it will also detect an ADET signal loss and will immediately act to swap pitch/yaw (P/Y) thrusters regardless of the actions taken in Celestial Sensor Logic. As pointed out before, this immediate action is necessary to prevent the possibility of extremely large angular excursions (>25 deg) from a thruster open failure.

It should be mentioned that the receipt of IDET or ADET loss power codes will cause CCS to initiate other necessary "safing" actions, such as the protection of instruments that may be damaged by direct exposure to the sun.

4.2 SUN SEARCH TURNS

Sun searches, performed via commanded turns, are a special case of all axis inertial turns, in which sun detection logic is employed to enable AACS to acquire the sun. An illumination detector (IDET) as part of the sun sensor is used to provide an indication that the sun is within a 40-degree by 40-degree FOV in pitch and yaw respectively. Two types of commanded turn sun searches are employed. The first type is used for the initial sun acquisition and for reacquisitions following a TCM, for which the CCS knows the attitude of the MM and can, therefore, send a predetermined set of fixed turns (P and Y). Subsequent sun searches, resulting from an inadvertent loss of the sun, are performed by CCS sending a series of P/Y commanded turns resulting in 4 steradian coverage.

For the reacquisition due to inadvertent loss, the mode switching logic is enabled during the turns such that when the IDET FOV is satisfied, the turn is automatically terminated and the subsequent sequence as directed above is used.

The mode switching logic (but not the sun detection logic) is disabled until the P/Y pair of commanded turns are complete and 30 seconds has elapsed. The switching logic is then enabled and if the sun is within the IDET FOV the mode is automatically switched to roll inertial such that the position error signal is received from the sun sensors (in P and Y) rather than the Dry Gyro Inertial Reference Unit (DRIRU). The sun sensors provide saturated output signals to drive the MM (using DRIRU for rate damping) until the sun is within the linear region of the sun sensor. The non-saturated sun sensor position signal plus rate estimation then drive the MM to the null position. During the search the roll control channel, driven by the roll error signals derived from the roll gyro, holds the roll attitude within the prescribed deadband. Upon completion of sun acquisition, Canopus acquisition is initiated by CCS.

4.3 CST LOGIC

When the loss of target is detected, a power code is sent to the CCS which responds to a series of reacquisition commands. These commands initiate a roll search.

4.4 CANOPUS ACQUISITION ROLL

Canopus acquisitions are performed by a special case or roll inertial mode, in which the CST provides the driving signal for the turn, in place of a turn commanded from CCS.

A search bias logic signal is sent to the CST from the HYPACE to position the scanned FOV to the positive roll error edge of the total FOV. Saturated negative output from the CST is summed with the roll rate signal derived from the roll gyro which causes the negative roll thruster to be actuated to establish the roll search rate. The CST logic software is enabled, and stars other than Canopus (or other selected stars) are discriminated against by the software on the basis of star intensity and cone angle. The HYPACE controls the CST cone angle via CCS command and the CST provides a star intensity signal to the HYPACE. When a star satisfying the intensity gate logic is detected, the CST demodulator is enabled by a logic signal from the HYPACE, and the CST begins tracking the star. The CST then provides a signal to the HYPACE software that is proportional to the roll angular error. The HYPACE software causes thruster actuation to achieve and maintain the sum of the CST roll error and roll rate signals within the deadband value. Canopus acquisitions are initiated by CCS command.

4.5 SEARCH RATES

Celestial reference search rates are 3.14 mr/s.

SECTION 5

SUBSYSTEM FUNCTIONAL REQUIREMENTS

5.1 INPUTS

For the sun sensor the inputs are sun sensor read and the sun sensor control command.

For the CST, the inputs which are required are the star sensor read (including the cone angle and intensity) and the CST control command.

SECTION 6

INTERFACE

Dual Sun Sensors. SS1 is dedicated to HYBIC 1 and SS2 is dedicated to HYBIC 2 as shown in Figure C5-6. No cross-strapping is provided. Since power to the SS is provided by the HYBIC, turning on HYBIC's J-supply will power up its associated Sun Sensor.

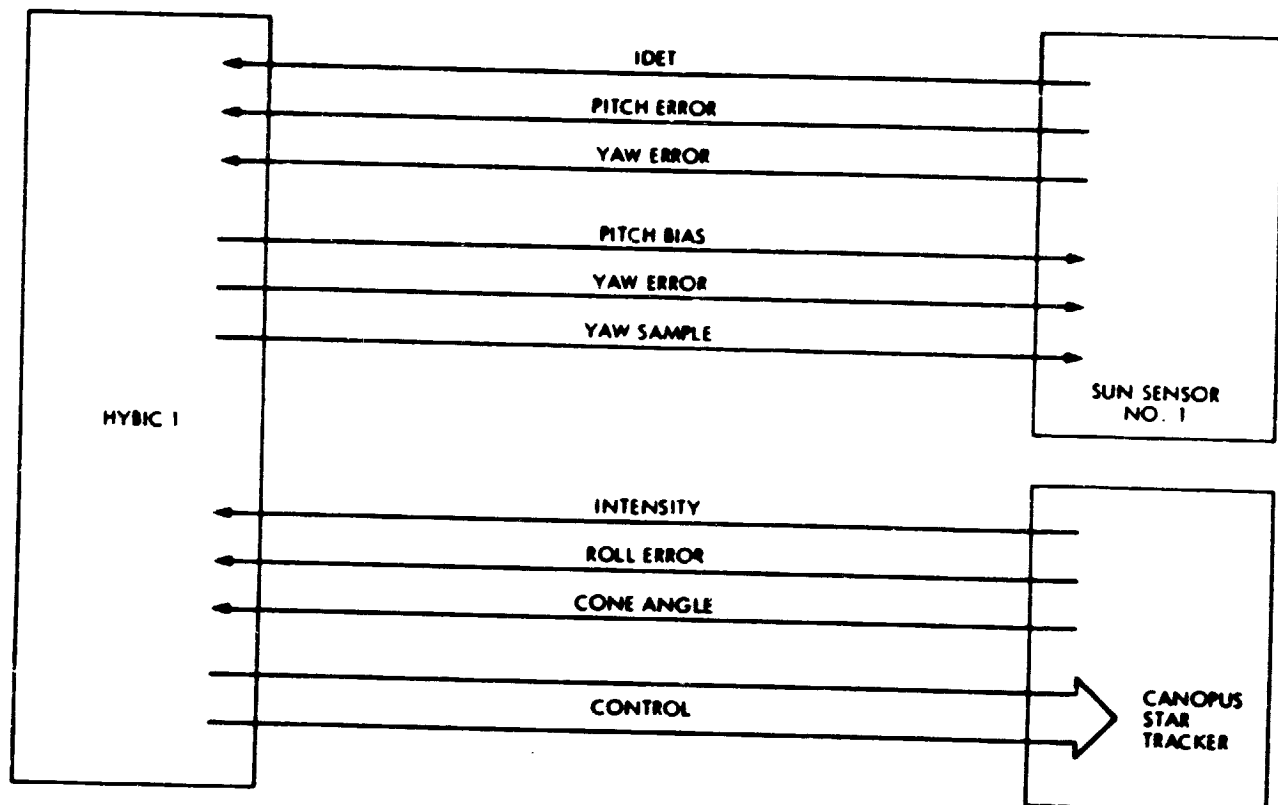


Figure C5-6. AACs/Sun Sensor and AACs/Canopus Tracker Interface

Dual Canopus Star Trackers. CT 1 is dedicated to HYBIC 1 and CT 2 is dedicated to HYBIC 2 as shown in Figure C5-6. No cross strapping is provided. Power to the CT is selected by either ground command or by a Power Code generated by the HYPACE software.

SECTION 7

PERFORMANCE REQUIREMENTS

The sensor logic is executed every 240 ms. The sun sensor and CST are read every 60 ms.

SECTION 8

IMPLEMENTATION CONSIDERATIONS

8.1 HARDWARE/SOFTWARE/RESOURCES

The sun sensor and CST logic use 274 AACS words. To process commands from CCS to AACS requires 69 words. The reads require 105 words.

8.2 JPL FLIGHT EXPERIENCE

Shortly after launch of Voyager 2 (the first spacecraft launched), bright particles, entering the field of view of the Canopus Star Tracker, caused flybacks and sweeps to recover Canopus reference. The repeated occurrence of this effect with the attendant Sun Sensor bias correction, caused a buildup of pulses. These pulses, which are used for attitude correction burns, exceeded established limits thus initiating a thruster branch swap.

This effect, called the "Bump in the Night", was corrected by using a slightly different approach to correct Sun Sensor biases. This problem, although corrected, was exacerbated by the fault protection and correction logic. More detection logic and better dynamic simulation would have helped to correct the problem during the design or prelaunch testing phase.

SECTION 9

VALIDATION

TBD

Appendix C

Section C6

Trajectory Correction and Attitude Propulsion

Unit (TCAPU) Fault Detection

DESCRIPTION OF AN AUTONOMOUS FUNCTION

SECTION 1

FUNCTION NAME

Trajectory Correction and Attitude Propulsion Unit (TCAPU) Fault Detection.

SECTION 2

FUNCTIONAL DESCRIPTION

This routine now in use on Voyager performs thruster pulse rate and angular position error checks to detect failures in the TCAPU thrusters, and contains the procedure for switching to redundant thrusters, and/or redundant Hybrid Buffer Interface Circuits (HYBICs) or Flight Control Programmers (FCPs).

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

The spacecraft must be able to maintain its attitude during all phases of the mission. Any single point failure in the attitude propulsion unit shall be detected and correction provided.

3.2 SYSTEM REQUIREMENTS

The spacecraft shall provide fault detection and correction algorithms to switch isolation valves in the event of a propulsion thruster failure. The Attitude and Articulation Control System (AACS) shall switch TCAPU's when abnormal leakage is detected or when high angular rates or large position excursions of the spacecraft are measured. The algorithm does not execute if the spacecraft is in either the Launch mode or the Propulsion Module mode since TCAPU thrusters are not used, or if a commanded turn of any type is in progress in which case the thruster pulsing rates can be quite large. The premise of the commanded turn itself is that the spacecraft's Z axis will eventually be pointed, not at earth, but in some other direction. Therefore, the primary objective of the fault detection/correction software, which is to maintain High Gain Antenna (HGA) boresight axis pointing at earth, would not be served by placing the commanded turn process within the purview of the TCAPU Fault Correction Control algorithm.

SECTION 4

SUBSYSTEM FUNCTIONAL OPERATION

As shown in Figure C6-1 the logic examines the possible operating modes; Celestial Cruise Maneuver (CC), Roll Inertial (RI), All-Axis Inertial (AAI), and Trajectory Correction Maneuver (TCM). Two of these modes, AAI and TCM, use gyro measurements to determine pitch and yaw angular position error. In the face of certain types of thruster failures which can cause spacecraft acceleration in these modes, it is necessary to examine the pitch and yaw angular position error to prevent very rapid loss of Earth-orientation. In the RI and CC modes, such a position error check is already provided in the Mode Monitor and Initialization routine based on the measurements obtained from the Sun Sensor; therefore, it is desirable to check angular position error only in the AAI and TCM modes as far as the TCAPU Fault Correction Control algorithm is concerned.

There are also a number of thruster failures which can also cause spacecraft drifts (i.e., at a nearly constant low rate) or abnormal operation along a deadband edge. In these cases, near-maximum duty cycle pulsing will occur on a thruster which is trying to counteract the effects of a "stuck-open" opposing thruster, or apparent near-maximum duty cycle pulsing from a "stuck-closed" thruster. Therefore, a pulse rate detection scheme is incorporated in the logic to distinguish such faults from the normal mode of thruster operation.

If either the angular position error or thruster pulse rate exceeds the prescribed limit, control is directed to the fault correction logic where the mode is again examined. If it is the TCM that has produced the fault, the TCM burn is immediately aborted by changing the mode back to AAI. It is assumed here that ground-based analysts can determine the cause of the problem and take appropriate action to correct it. For any other mode, the logic proceeds to isolate the fault with regard to roll axis control or to pitch/yaw axis control. With this accomplished, it is determined whether previous faults have been detected in this axis, and if not, the backup thruster set (either pitch/yaw or roll) is swapped for the set in use. If a swap of redundant thruster sets has already taken place, it is assumed that this is a "persistent" fault which must have occurred in the thruster interface in the HYBIC. A swap of redundant HYBICs is then initiated by calling the Catastrophe Handler subroutine.

After proper corrective action has taken place, the process of recovery from the fault begins. Large angular errors may persist or additional large angular errors may persist, or additional large pulse rates may be required to recover; therefore, the algorithm increases (doubles) the angular error and pulse rate limits after initiating corrective steps. Later, after the situation has been stabilized, the original limits can be restored by ground command.

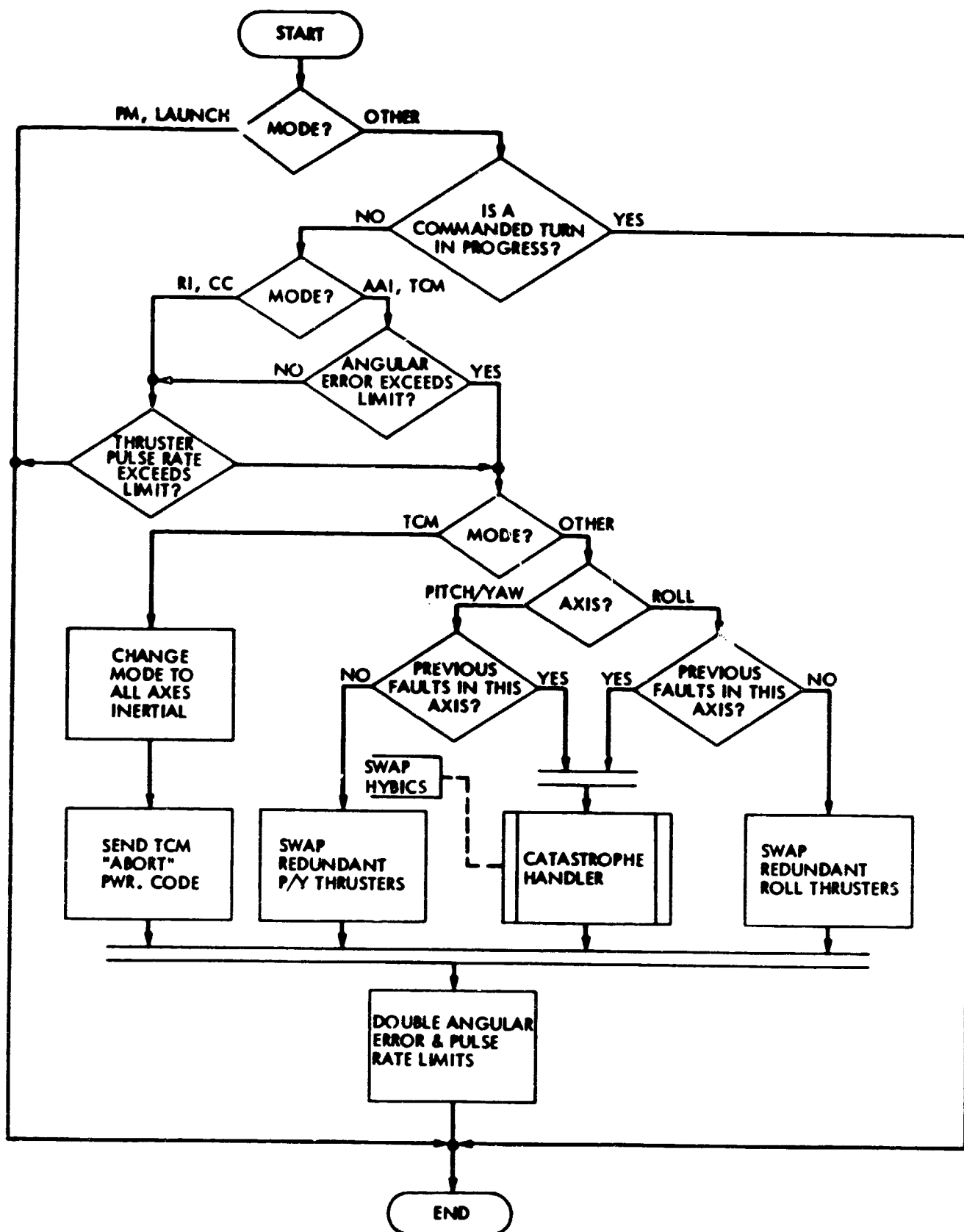


Figure C6-1. TCAPU Fault Correction Control Logic

SECTION 5

SUBSYSTEM FUNCTIONAL REQUIREMENTS

5.1 INPUTS

Entries to this routine are (1) enable flag, (2) wait flag, (3) turn flag, (4) turn timer and counters, and (5) mode flags. These flags are set by the Computer Command System (CCS). Ground command can also set the turn flag and turn timer and counter.

5.2 PROCESSING

This routine can be enabled or disabled only by direct CCS command. As shown in the flow diagram of Figure C6-2, it will also be temporarily disabled while a HYBIC swapping operation is in progress.

Basically, two types of tests for thruster failure are performed in this algorithm. One is a test for spacecraft attitude (angular) error; the other is essentially a thruster pulse rate test. Each type of test is performed for each control axis; pitch, yaw, and roll.

To provide data for the pulse rate test, the control law (60 ms) subroutine will increment a counter each time it decides that a (+) or (-) thruster is to be open during that particular 60-ms control computation interval. In the usual cruise condition, TCAPU thrusters are permitted to be open only 10-20 ms during the 60 ms control computation interval, and the pitch, yaw, and roll "open" periods are staggered within that 60 ms interval to save power. Under these conditions, the count will truly represent pulses, i.e., transitions from "closed" to "open" valve states. However, during spacecraft turn maneuvers and certain situations in which large maneuvers are needed to reacquire Sun or star references, thruster power sharing is disabled and all thrusters are allowed to remain open throughout the 60-ms interval. In these cases, pulse count does not necessarily reflect valve state changes, but simply total "open" time. Of course, it is also possible that a pulse count represents only the attempt to open a thruster that may have failed closed. Conversely, no pulse count will be directly produced in case of a thruster open failure, but presumably the opposing thruster will be commanded to open in an attempt to maintain control and thereby produce an unusual pulse count.

By periodically resetting to zero the pulse counts thus obtained for each thruster, the test becomes one of monitoring pulse rate. The period selected for pulse-count accumulation is 5 min. The pulse-count limit for that period is set at 750 pulses. Since the maximum number of pulses possible (per thruster) in 5 min is 4800, the limit represents 16% of the maximum pulse rate. This was designed to prevent certain relatively high-rate but normal, pulsing maneuvers, such as Sun sensor bias insertion (or removal) from tripping the limits, but at the same time to distinguish serious thruster problems.

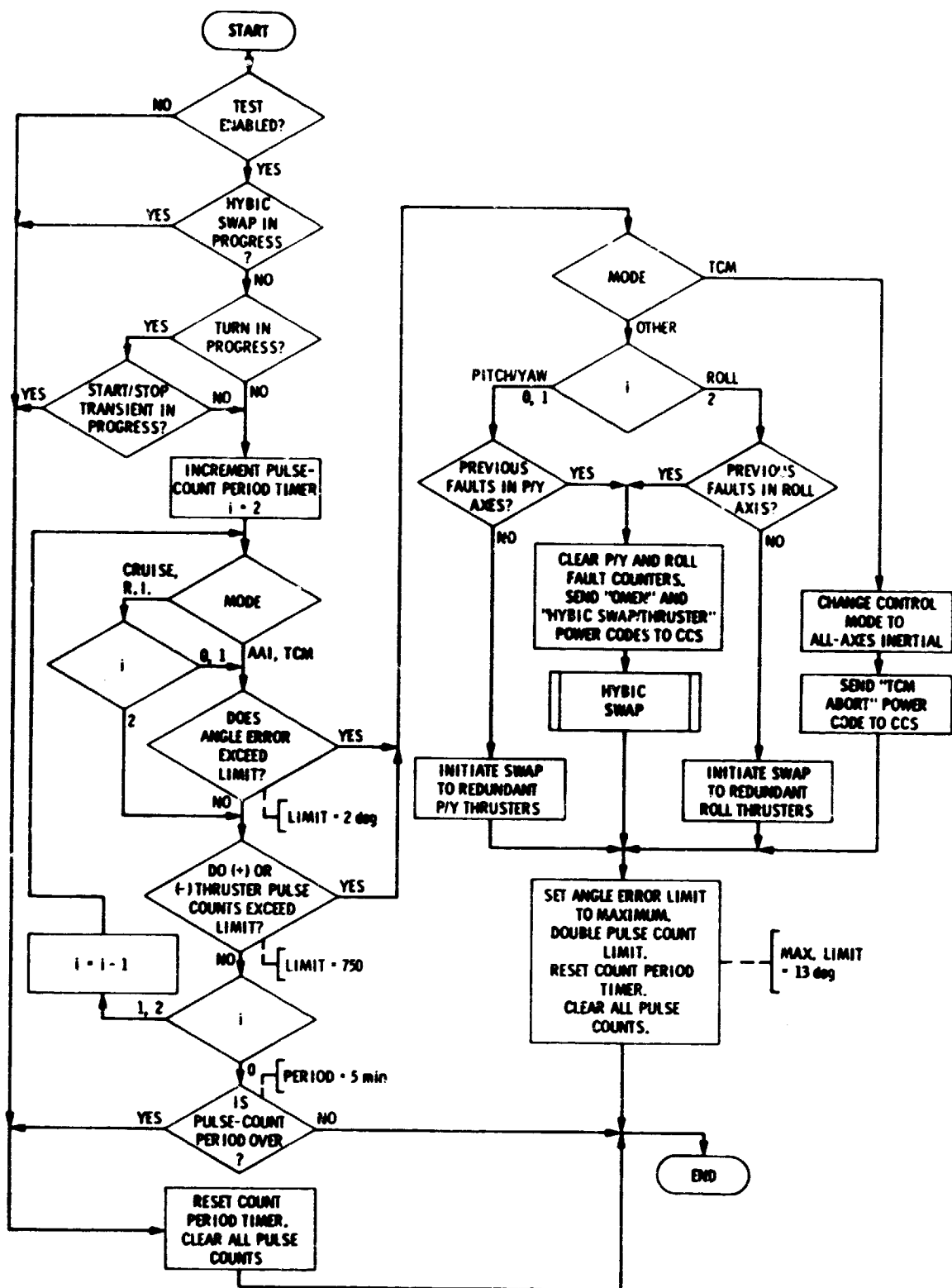


Figure C6-2. TCAPU Fault Correction Control Routine

Generally speaking, the pulse-rate test is designed and best suited for identifying closed failures or extremely degraded thruster output cases. In a cruise situation, if the thruster cannot reverse the vehicle angular rate at the control deadband edge, the pulse-count limit will be exceeded in about 45 s. At a cruise limit cycle rate of 10 rad/s, this means finding the fault before an angular excursion of 0.026 deg beyond the deadband. Since a great deal of importance is attached to maintaining Earth-spacecraft communications through the high-gain antenna, limiting angular excursions due to failures is particularly critical in pitch and yaw, since the roll axis is the antenna boresight axis. A partial-open failure (i.e., a large leak) is also well suited to the pulse rate test, where the opposing thruster is able to maintain angular error control, but is required to expend relatively large amounts of propellant.

On the other hand, in a total or near-total open-failure condition where the opposing thruster is unable to effectively counteract the failure (particularly if it can only pulse for 10-20 ms), the angular error will quickly increase to a very large value before a pulse count limit of 750 could initiate a timely correction. In such a situation, the only effective method of detection is an absolute angular error limit. For this purpose, a limit of 2 deg was chosen, primarily because it is just below the saturated output level of the Sun Sensors' angular error detector.

Figure C6-2 indicates one additional circumstance that can temporarily disable TCAPU fault testing during a turn maneuver's starting or stopping transient. A relatively large number of pulses are produced during these transients that could falsely trigger the fault detectors. Thus, for 30 s at the start and 60 s at the end of a turn, the tests are bypassed. However, during the cruising portion of the turn (some roll turns are hours long), fault testing is carried out as usual. Figure C6-2 also shows that the pulse rate test is performed for all axes in all control modes, but the angle limit is not used for the roll axis during the Roll Inertial or Celestial Cruise control modes. This is due to the likelihood of undesirable interactions with the Canopus Star Tracker (CST) logic, particularly the "flyback-and-sweep" feature used to locate a star within its +3.5 deg field of view. Such an action, which is quite common in those modes, could needlessly trigger the 2 degree angle limit. While this leaves the roll axis somewhat more vulnerable to roll thruster open failures, the impact on communications is not serious.

When a fault is detected, it is necessary, because of the TCAPU arrangement of latching isolation valves (see Figure C6-3), to pinpoint the problem in either a pitch/yaw thruster group or a roll thruster group. Action can then be taken to command a change in isovalve states to close off the offending group (in either Branch 1 or Branch 2) and activate the redundant group of thrusters in the other branch. However, in the case of a trajectory correction maneuver (TCM), which may employ either the 4 TCM thrusters alone or in addition to the other 6 thrusters, a fault detected in pitch or yaw would not necessarily indicate whether a TCM thruster group or the other P/Y group was at fault. Therefore, in the TCM mode, the action taken in response to any TCAPU fault is simply to abort the maneuver by closing the TCAPU isolation valves. Ground controllers can then analyze the telemetered results and take corrective steps at a later time. Notice that if a failed TCM thruster is discovered, it can be isolated from the system and subsequent maneuvers can still be accomplished using a half-system.

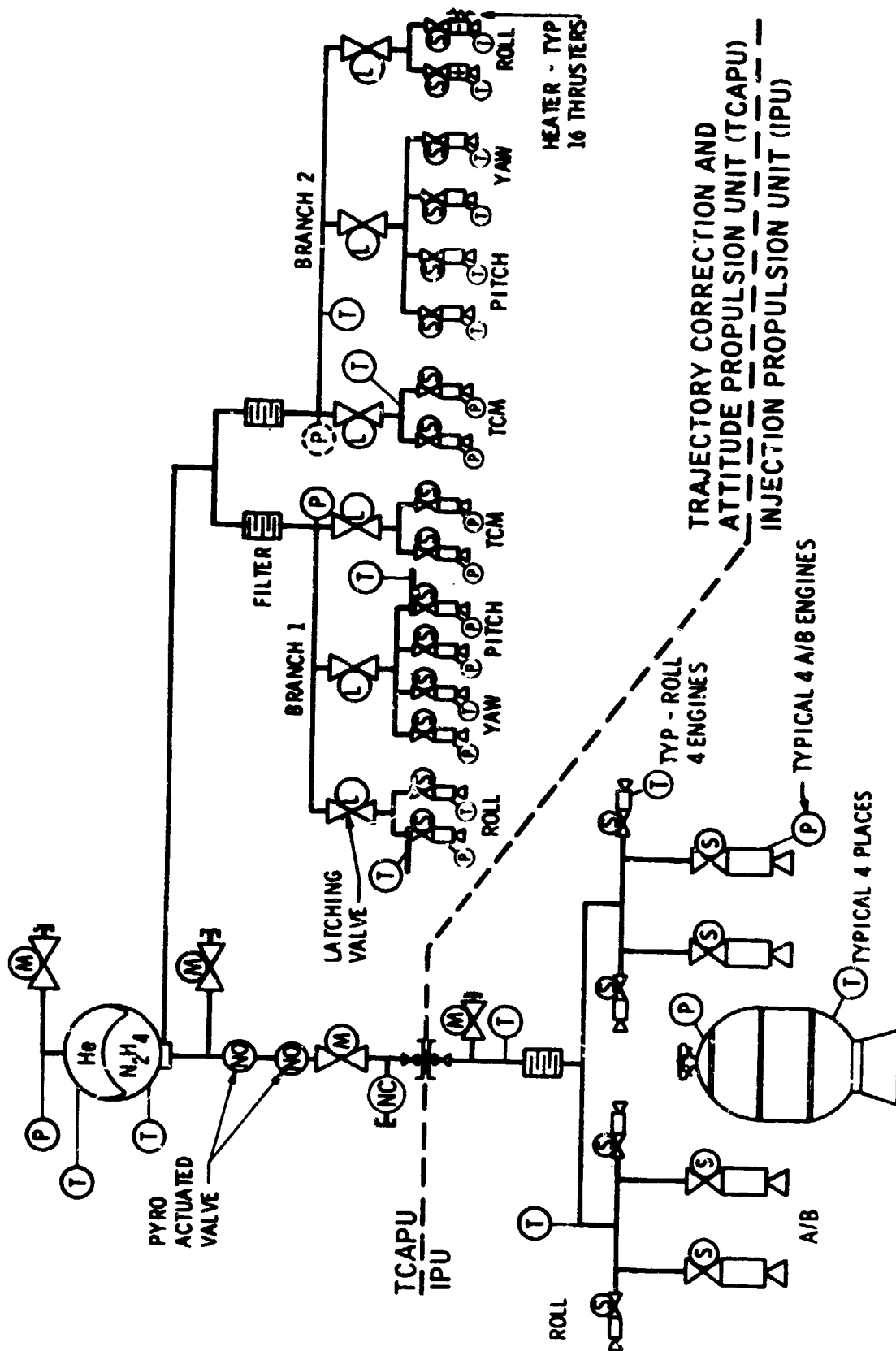


Figure C6-3. Voyager Propulsion Subsystem

Having detected a problem and taken action to correct it, the algorithm will reset the angle limit to 13 deg, which is near the maximum angular error that can be held in an 18-bit memory word at a scale factor of 0.357 arc-sec/bit, and double the pulse-count limit to 1500. These two actions provide the attitude control system with sufficient margin to recover from a real thruster failure transient. Also, the test timer and accumulated pulse counts are reset to zero.

If, by chance, swapping to redundant thrusters does not cure the problem, another error limit violation in the same area (P/Y or Roll) will cause a HYBIC swap to be executed. Again, as in the case of repeated "gyro faults", the suspicion is that some portion of the interface hardware must therefore be the source of trouble. In that case, an "Omen," followed by a special diagnostic power code, will be sent to CCS. If a HYBIC swap is triggered here, all memory of previous TCAPU plumbing swaps is erased, although the plumbing will not be simultaneously swapped back again. It will, of course, be up to ground controllers to restore test limits to their normal values after any TCAPU fault corrections.

5.3 OUTPUTS

Angular and pulse limits are output to the gyros logic, to the thrusters and the CCS. Upon detection of faults, the following power codes may be sent: PC044, PC064, and PC066. When, as a result of a pulse test failure in TCAPU during the TCM, this maneuver is aborted and the mode is returned to all axis inertial with appropriate initialization.

SECTION 6

INTERFACE

Figures C6-4, C6-5, and C6-6 show the interfaces with the isolation valves, the pitch and yaw TCAPU valves and the roll TCAPU valves.

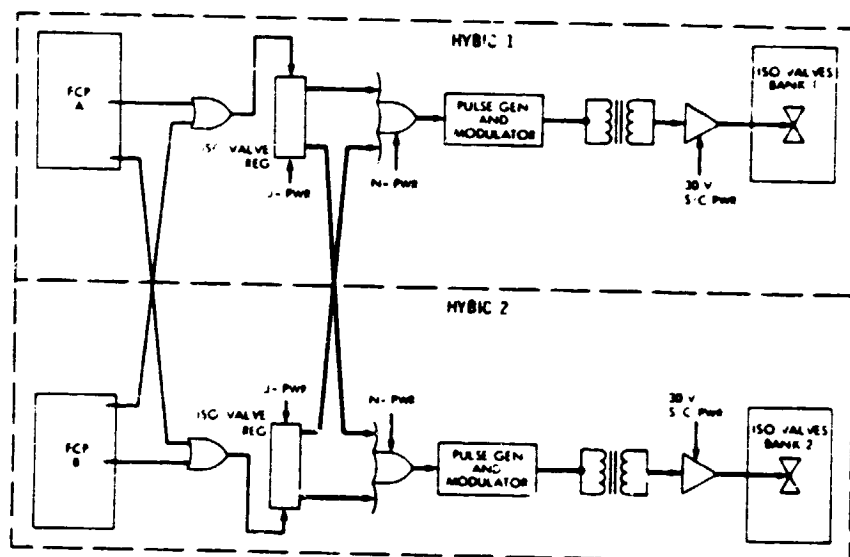


Figure C6-4. Isolation Valve Cross-Strapping

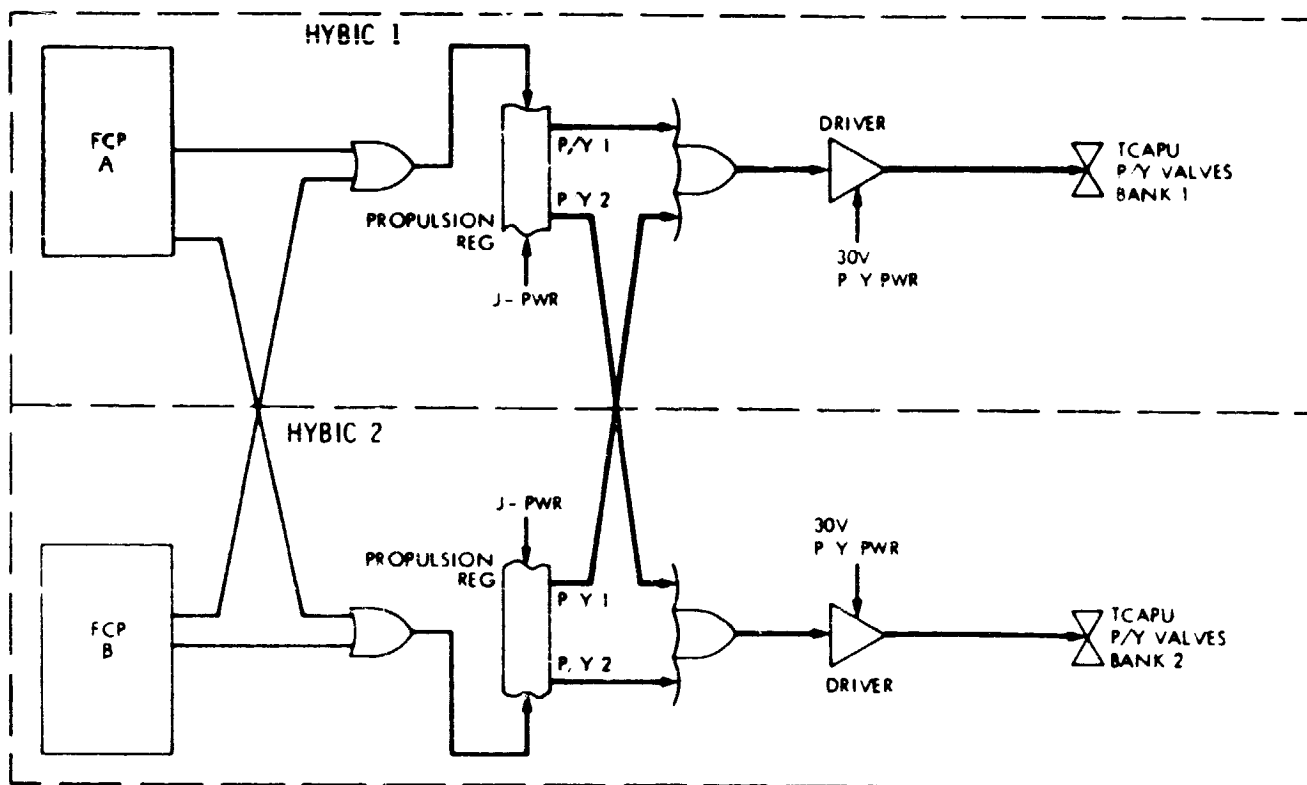


Figure C6-5. Pitch and Yaw TCAPU Valve Cross Strapping

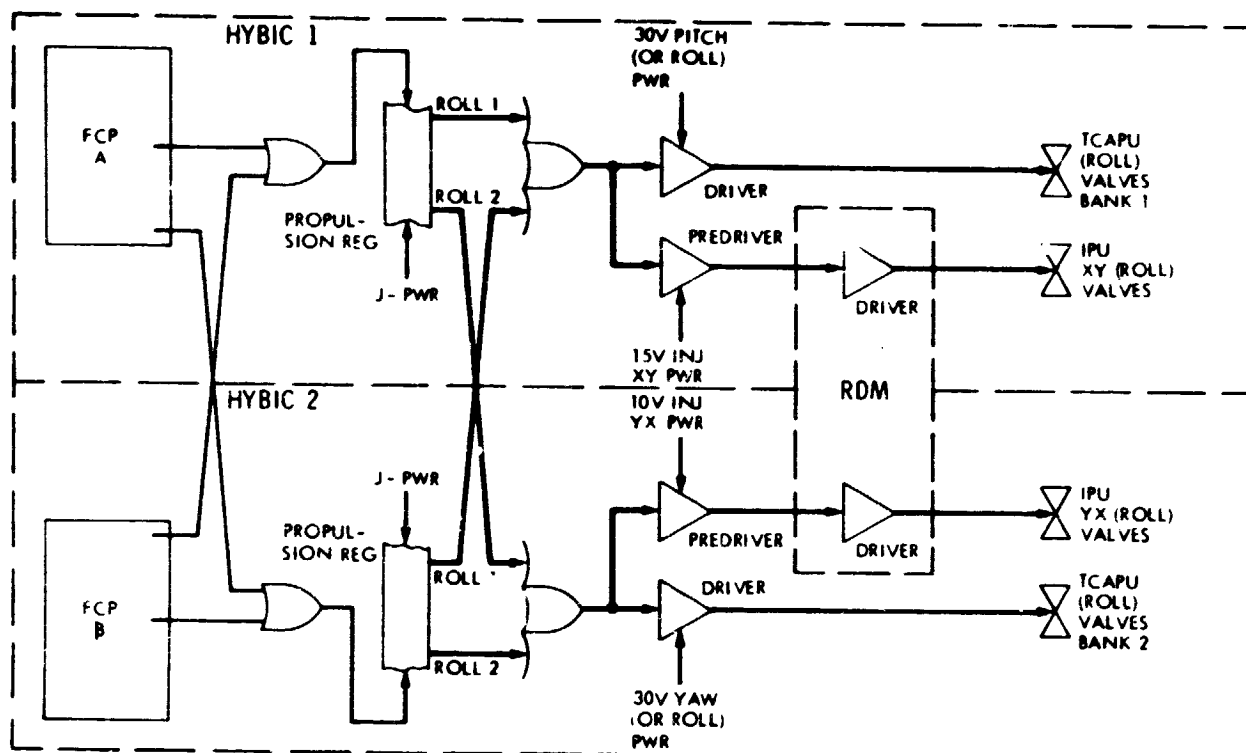


Figure C6-6. Roll TCAPU Valves and IPU Valves Cross Strapping

SECTION 7

PERFORMANCE REQUIREMENTS

This routine is performed every 240 ms except during the Launch phase and HYBIC swaps.

SECTION 8

IMPLEMENTATION CONSIDERATIONS

8.1 HARDWARE/SOFTWARE/RESOURCES

This routine requires 97 words in the AACs.

8.2 JPL FLIGHT EXPERIENCE

Several unexpected problems with the S/C hardware and software caused the swap of thruster branch plumbing and subsequent circuitry swapping in an attempt to correct the problems.

These problems were:

- Propulsion module separation - A planned isovalve reset delayed thruster firing causing a buildup of attitude errors.
- "Bump in the night" - Repeated flybacks and sweeps to recover Canopus reference caused a buildup of pulse counts for attitude correction.
- Pitch, turn, overshoot and high duty cycle oscillations - Narrow deadbands and thruster plume impingement caused attitude errors during certain turn maneuvers.
- Magnetometer Boom Deployment - Unexpected spacecraft rates due to boom deployment caused a buildup of pulse counts in attitude correction.
- Sun Sensor Transient - A temporary pitch sun sensor anomaly induced an unacceptable attitude error.

Command Error

- A propulsion system pressure loss combined with thruster plume impingement degraded thruster performance sufficient to cause a buildup of pulse counts above limits.

The spacecraft recovered from these events using the fault correction software routines by returning to a safe, stable state. The problems mentioned above were corrected by widening deadband, increasing limits, and/or inhibiting the thruster fault routine during transient effects.

SECTION 9

VALIDATION

TBD

Appendix C
Section C7
TRNSUP Routine

SECTION 1

FUNCTION NAME: TRNSUP

SECTION 2

FUNCTIONAL DESCRIPTION

The purpose of the TRNSUP routine is to provide a means of verifying the integrity of the Computer Command Subsystem (CCS) and Attitude and Articulation Control Subsystem (AACS) prior to the execution of a critical on-board function--typically the spacecraft attitude-modifying activities associated with a propulsive trajectory correction maneuver (TCM). The use of the tandem check capability of this routine requires that both halves of the CCS be functioning properly. In the event of a failure of one half of the CCS, or a decision to not use both halves, TCMs may be executed without use of the tandem check function.

2.1 TRNSUP INTERFACES

Figure C7-1 illustrates the TRNSUP inputs, processing and outputs.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

The basic requirement for the existence of a routine of this type is that a TCM shall be executed within certain limits or not executed at all. Also, the criticality of a TCM is such that all resources available shall be used to insure its proper execution. Therefore, both halves of the CCS shall be used to implement the tandem check function within TRNSUP.

3.2 SPACECRAFT REQUIREMENTS

In support of the basic requirement stated above, the following spacecraft requirements shall be implemented.

3.2.1 Health Check

The ability to verify basic health of the primary executors of the TCM (CCS and AACS) before actual execution.

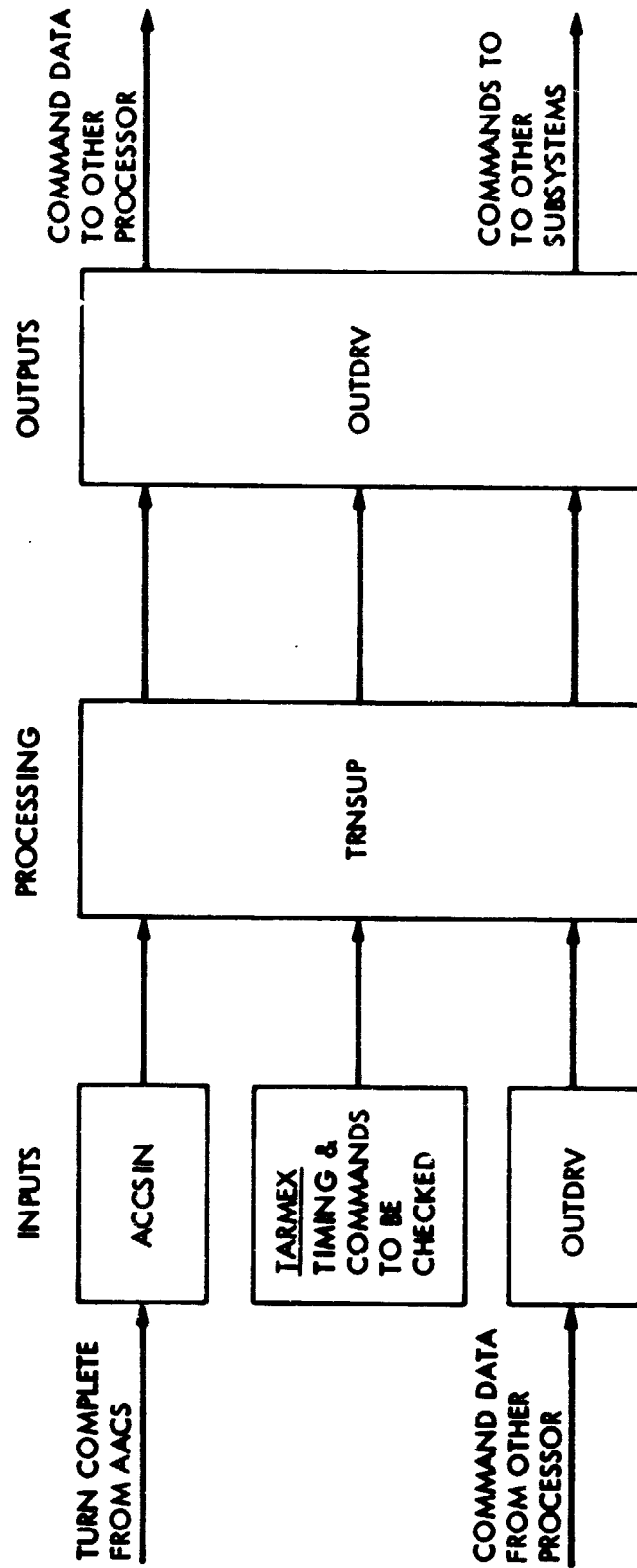


Figure C7-1. TRNSUP Interface Block Diagram

3.2.2 Tandem Check

The ability to utilize both halves of a redundant subsystem (CCS) to validate the correctness of the critical commands issued by that subsystem during execution of a TCM.

3.2.3 Commanded Turn Verification

The ability to verify the duration of commanded turns as executed by the AACS.

3.2.4 Spacecraft Reorientation

The ability to direct the re-orientation of the spacecraft back to the sun and earth if anomalous operation is detected after the start of the spacecraft attitude changes that precede the motor burn.

SECTION 4

SUBSYSTEM FUNCTIONAL OPERATIONS

The Tandem and Turn Support Routine (TRNSUP) is optionally employed whenever a spacecraft sequence is to be executed that requires maneuvering away from celestial references or includes a propulsive trajectory correction event. TRNSUP is loaded with the sequence as a utility routine and is called by the executing sequence to perform the following functions:

- (1) To issue CCS "tandem" events.
- (2) To check key fault indicators as a go/no-go test for subsequent sequenced events.
- (3) To check for proper maneuver turn durations.

4.1 TANDEM EVENTS

Tandem events issued by the CCS require that both CCS processors agree on the timing (within 900 msec) and content of the command data bits to be issued to the receiving subsystem (usually AACS). If either criterion is not met, the command is not issued, the executing sequence is halted, and a safing sequence is called. The function of the safing sequence is to assure that subsequent recovery data are recorded on-board, and that the spacecraft reacquire its celestial references. The two halves of the CCS execute the tandem check process in a master/slave configuration. The 'master' processor will output a command only if it agrees with the command data bits sent to it from the 'slave' processor. The slave processor will not output the command to the affected subsystem, but will initiate an 'abort' sequence if it does not agree with the data bits sent to it from the master processor.

4.2 CHECKING KEY FAULT INDICATORS

Whenever it is desired to check the status of fault indicators stored in CCS prior to executing an event, TRNSUP offers the option for the sequences to test for (a) prior celestial reference loss, (b) CCS tolerance detector trip status, and (c) error-indicating power codes from AACS.

If a prior reference loss has occurred, the sequence is terminated. If either a tolerance detector trip indication or an error-indicating power code trace is present, the sequence is terminated and the safing sequence is executed.

4.3 MANEUVER TURN DURATION

One final capability that the TRNSUP affords is checking the duration of maneuver turns. The sequence can be designed to call TRNSUP with a "turn window open" and a "turn window close" event. If TRNSUP determines that the TURN COMPLETE power code from AACS has been received at the window open time (too short a turn) or has not been received at window close time (too long a turn), the sequence is terminated, a turn abort command is issued to AACS, and the general safing routine is executed.

4.4 FUNCTIONAL DATA FLOW DIAGRAM

The functional data flow diagram is shown in Figures C7-2 and C7-3.

SECTION 5

SUBSYSTEM FUNCTIONAL REQUIREMENTS

5.1 INPUTS

Inputs to the routine shall consist of 'calls' which shall request a particular health check to be executed, and data which are the actual commands to be validated before execution. These calls and data shall originate within other software (TARMEX) in the CCS. These calls shall also determine whether the routine is to be used as the master or the slave when validating a command for output to another subsystem.

The OUTDRV routine shall fetch and make available to the TRNSUP routine the other processor's command data for validation.

This routine shall also receive (via AAC SIN) a turn complete signal from AACS. This shall be used to verify the correct turn duration as executed by the AACS.

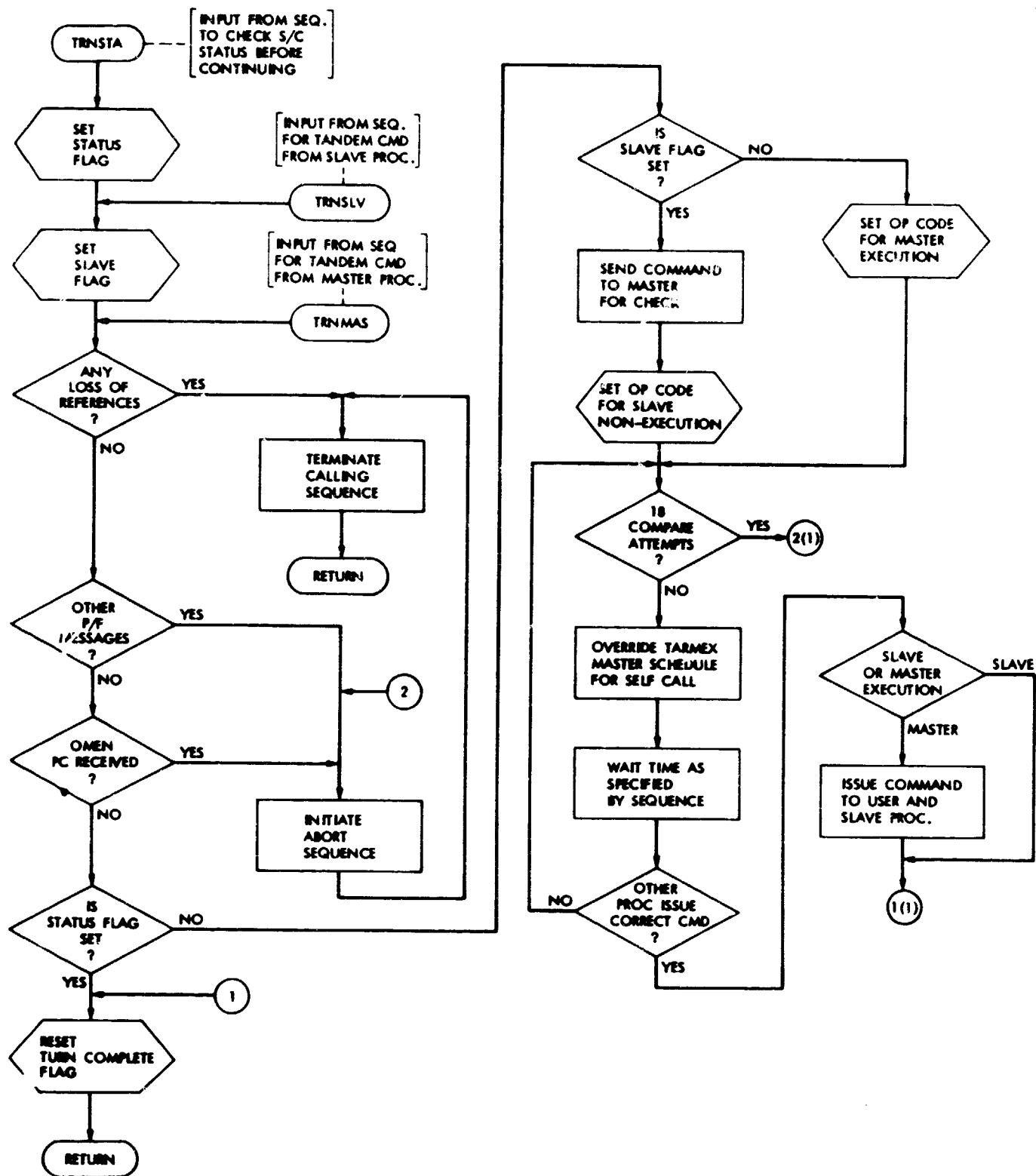


Figure C7-2. Flow Diagram, TRNSUP Routine

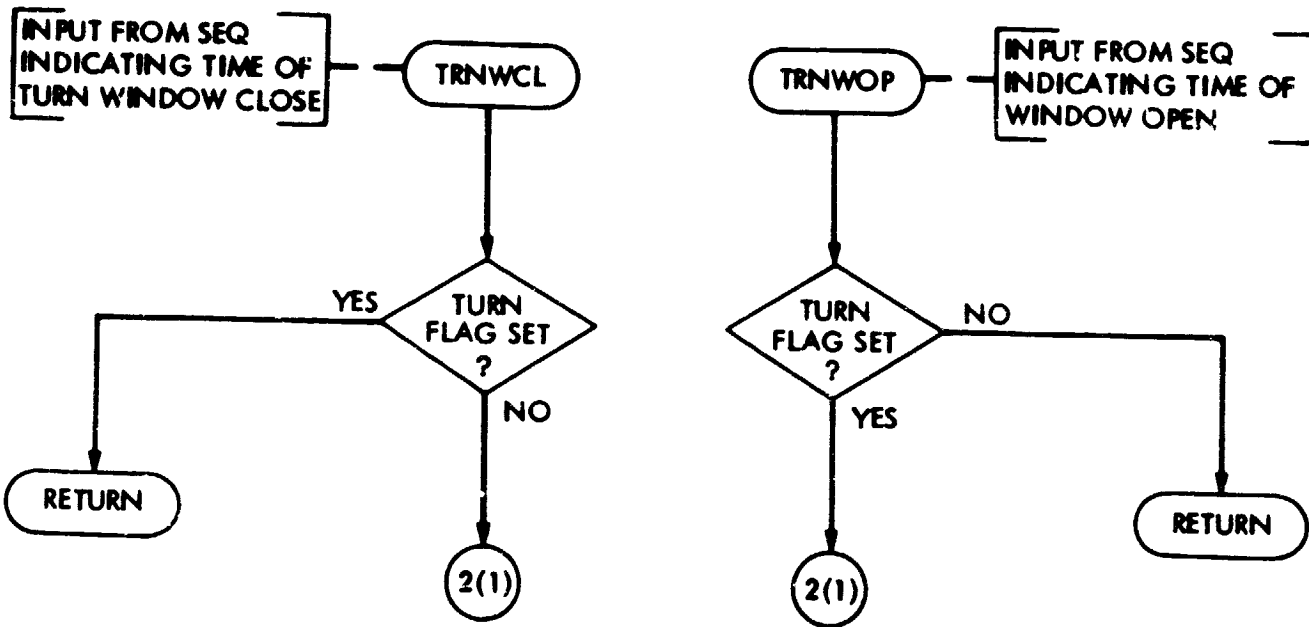


Figure C7-3. Flow Diagram, TRNSUP Routine

5.2 PROCESSING

5.2.1 Initialization

This routine shall be entered at several different times during the execution of a commanded turn maneuver. The first entry (TRNSTA) shall occur before a turn start, and shall serve to check the condition of the AACS and initialize the routine. The condition of the AACS shall be that proper star and sun acquisitions be in effect, and that no "omen" power code has been received. If either of these conditions exist, the maneuver routine shall be terminated. In addition, if an "omen" power code has been received, or if the other CCS processor has had a tolerance detector trip without this processor experiencing one, the abort sequence of Table C7-1 shall be executed. The abort sequence shall not be executed for a loss of acquisition because the normal CCS response to an acquisition loss will accomplish approximately the same thing. Once an "omen" or loss of acquisition has been received by the CCS, ground intervention shall be required to remove the indication from the CCS.

5.2.2 Tandem Command Check

After the initial AACS status check and routine initialization, the maneuver execution shall proceed. If the maneuver is to be performed using the tandem check capability, entry for each tandem command shall be at either the master entry point (TRNMA) or the slave entry point (TRNSLV). Either entry shall cause the AACS status and the other CCS processor's tolerance detector to be rechecked. Next, the routine shall prepare for the tandem check process by setting up for a maximum of eighteen 50-millisecond tests for a tandem compare. The slave processor shall then output the tandem command to the master processor for comparison. Then each processor shall reduce a tandem test-count word by one. The routine shall force the sequence support routine to stay in the multiple-event mode and overlay the pointer to the command being checked with a point to the check sequence. Both processors shall wait for the next check-time (50 msec) determined by the overlaid multiple event block currently active in the sequence support routine. At this point in time, the master shall be waiting to compare its command with that transferred from the slave, and the slave shall be waiting for the master to send a command for it to compare.

If the master receives and agrees with the command transferred from the slave within eighteen checktimes (900 msec), it shall output the command, first as a non-execute command to the slave processor, and then as an execute command to the AACS or other user. The slave processor shall then compare the command it received from the master with the command it originally sent, and if they agree, it shall allow the sequence to proceed. If either processor disagrees with the other, or the 900 milliseconds elapses, the abort sequence of Table C7-1 shall be executed.

Table C7-1. Maneuver Abort Sequence

- 1. DSS ON**
- 2. DSS RECORD 7.2 KBPS OR NO COMMAND***
- 3. FDS 1200 BPS, TCM TELEMETRY MODE**
- 4. TURN ABORT**
- 5. ALL AXES INERTIAL**
- 6. DEADBAND UPDATE**
- 7. DISABLE ROLLBACK**
- 8. WAIT 6 MINUTES**
- 9. SUN SENSOR SEARCH ENABLE**
- 10. WAIT 1 SECOND**
- 11. START CANOPUS SEARCH**

*** DEPENDENT ON MISSION PHASE.**

After each processor has finished processing the tandem command, the routine shall restore the multiple-event pointer for the tandem sequence in sequence support routine and return to that routine.

5.2.3 Turn Stop Check

Because the AACS determines when a commanded turn shall end, the CCS shall verify that the end occurred within a time window. The maneuver sequence shall therefore open and close that window.

The window shall open by a transfer to the TRNWOP entry point. This entry point shall check an indicator word which shall be all zeros if the turn-complete power code has not been received (should not have been at this time). The window shall close by a transfer to the TRNWCL entry point. At this time the indicator word should be non-zero, and if it is the maneuver shall be allowed to proceed. If the indicator is wrong at either check, the abort sequence of Table C7-1 shall be executed.

5.2.4 Constraints

- (1) This routine shall be entered from a multiple-event pseudo-event from the sequence support routine only.
- (2) The specified time spacing between commands in that multiple-event block shall be in the centisecond time-base.
- (3) All commands to be checked in the tandem mode before execution shall be assembled in the multiple-event block with Bit 1=0. Additionally, DC commands must have Bit 2=1 also.
- (4) Prior to the actual commands or turn duration to be checked, this routine shall be entered at (transferred to) TRNSTA to initialize the routine and verify AACS status.

5.3 OUTPUTS

This routine's outputs shall be determined, in part, by its use as either the 'master' or the 'slave' when command validation before execution is utilized. If used as the master, this routine shall output a command, originally input from other software, to the required external subsystem, typically AACS, and to the other half of the CCS (both via OUTDRV). If used as the slave, this routine shall only output the command to be validated to the other half of the CCS.

SECTION 6

INTERFACE LIST

A listing of the external interfaces is shown in Table C7-2.

SECTION 7

PERFORMANCE REQUIREMENTS

A valid tandem-check comparison of TRNSUP shall occur within 900 milliseconds of a TRNMA or TRNSLV entry.

SECTION 8

IMPLEMENTATION CONSIDERATIONS

8.1 GENERAL DESCRIPTION

This routine was modeled after a similar routine in Viking. However, the Viking routine required that all commands associated with a maneuver from start to finish be in the tandem mode. For normal TCMs this was acceptable even though some of the commands were not critical. For Voyager, it was desired to use the tandem-check capability for turn commands associated with a science (non-propulsive) maneuver as well as TCMs. Science maneuvers typically have many non-critical commands. Because more CCS memory is required to issue commands in the tandem mode, Voyager required that this capability be implemented on a selective basis.

Regardless of the method used, the use of redundant halves of the CCS for output validation necessitated that the hardware design provide for a data interchange between halves. Without this hardware capability, command validation by TRNSUP could not be accomplished.

8.2 RATIONALE FOR TECHNIQUES USED

With implementation of a selected tandem-check function as opposed to the Viking all-or-nothing approach, the tandem-check maximum time went from a hardware-controlled 160 milliseconds to a software-controlled 900 milliseconds. This necessitated the use of the timekeeping function available within the TARMEX routine for determination of the maximum check time of 900 msec. The 900 milliseconds was selected because there could be up to 500 milliseconds of time skew between halves of the CCS when using this software scheme. The Viking hardware approach was not affected by this skew.

Table C7-2. Interface List, TRNSUP Routine

FROM/TO	WHAT	HOW (MEDIUM)	INTERFACE DOCUMENT REFERENCE
AACSIIN/TRNSUP	TURN COMPLETE	FROM AACCS SUBSYSTEM	PARA. 5.1; FIG. 2.1-1
TARMEX/TRNSUP	TIMING	CENTISECOND TABLE	PARA. 5.1; FIG. 2.1-1
TARMEX/TRNSUP	COMMAND DATA	SEQUENCE TABLE	PARA. 5.1; FIG. 2.1-1
OUTDRV/TRNSUP	OTHER PROCESSOR DATA	HARDWARE INTERFACE	PARA. 5.1; FIG. 2.1-1
TRNSUP/OUTDRV	COMMAND DATA TO OTHER PROCESSOR	TRNSUP PROCESSING	PARA. 5.3; FIG. 2.1-1
TRNSUP/OUTDRV	COMMANDS TO OTHER SUBSYSTEMS	TRNSUP PROCESSING OR ABORT TABLE	PARA. 5.3; FIG. 2.1-1

8.3 FLIGHT MODIFICATIONS

None.

8.4 IMPACT OF NEW TECHNOLOGY

This routine was necessary because the host computer contains no inherent capability to validate outputs before transmission. A software tandem check function such as this routine typically would not be required where the host computer had fault tolerance built into its basic architecture such as proposed for the Redundancy Management Subsystem (RMS) for DSCS III. However, the desire to check the status of other subsystems prior to issuance of a critical command may still be a software requirement.

8.5 RESOURCE REQUIREMENTS

The TRNSUP routine requires 68 words of CCS memory. Each entry at TRNSTA requires 2 words within sequence memory. Each entry at TRNMAS or TRNSLV requires 4 to 8 words of sequence memory depending on command type.

SECTION 9

VALIDATION AND TEST REQUIREMENTS

Validation and test of the TRNSUP routine shall be accomplished at two levels: subsystem and system.

9.1 SUBSYSTEM VALIDATION

At the subsystem level, verification of the coded routine's ability to satisfy the specified requirements shall be accomplished by simulation of the execution of the routine on a test computer rather than the host computer. This shall allow easy variation of the input calls and data to validate proper operation of the routine.

As in any software system, individual routines or software modules are designed to operate in conjunction with other software routines or modules. Consequently the TRNSUP routine shall utilize the following routines during simulation testing at the subsystem level:

9.1.1 TRNSUP Activation

TARMEX shall provide the time-separated inputs to activate TRNSUP.

9.1.2 Timing Inputs

TRAPS, COINTS shall provide timing inputs to TARMEX so that it can output the time-related calls to TRNSUP.

9.1.3 Command Data Transfer

OUTDRV shall provide the other processor's command data to TRNSUP and shall output TRNSUP commands to the other processor and subsystems.

9.1.4 Attitude Control Inputs

AACSIN shall provide the Attitude Control inputs for TRNSUP (AACS health and turn complete signals).

9.1.5 Fixed Constants

GLBCNT shall provide the fixed constants used by TRNSUP during execution.

9.1.6 Variable Memory

VARABL shall provide the variable or 'scratchpad' memory required by TRNSUP.

9.2 SYSTEM (SPACECRAFT) LEVEL VALIDATION

TBD.

Appendix C
Section C8
ERROR Routine

SECTION 1

FUNCTION NAME: ERROR

SECTION 2

FUNCTIONAL DESCRIPTION

The ERROR routine is included within the Computer Command Subsystem (CCS) software system to respond to anomalous CCS hardware and software conditions. It typically puts the CCS in a known, quiescent state and waits for ground action.

2.1 ERROR ROUTINE INTERFACES

Figure C8-1 illustrates the ERROR routine's inputs, processing and outputs.

SECTION 3

GENERAL REQUIREMENTS

3.1 MISSION REQUIREMENTS

The CCS on board the Voyager spacecraft provides the command decoding and distribution function. Because of the critical nature of this function, both halves are powered and configured to receive ground commands at all times. Therefore, the function of the ERROR routine shall be to minimize the likelihood of a processor outputting erroneous commands or interfering with the other processor's ability to output valid commands in the event of an anomalous condition within a processor.

3.2 SPACECRAFT REQUIREMENTS

Upon sensing an anomalous condition within a processor, the ERROR routine shall cease any activity in progress and revert to the 'wait' state. In addition, an indication of the type of anomalous activity shall be stored within the CCS memory to aid in anomaly determination and recovery.

SECTION 4

SUBSYSTEM FUNCTIONAL OPERATION

Whenever an abnormal condition in either hardware or software exists within the CCS, the ERROR routine is entered. The response generally is to place the CCS in a known, quiescent state. Upon entry, the routine determines the source of entry and stores:

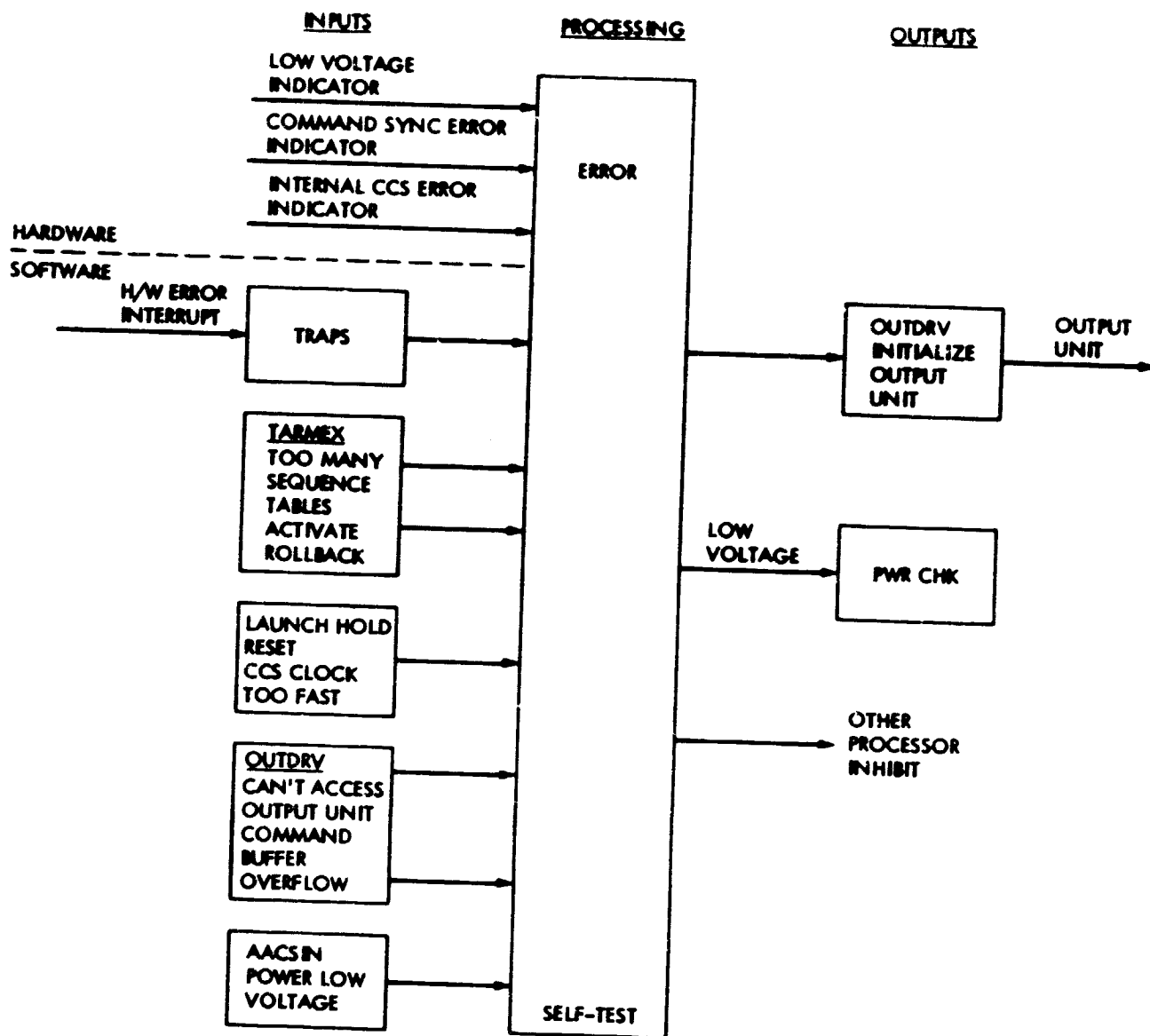


Figure C8-1. ERROR Interface Block Diagram

- (1) The error condition.
- (2) The value of its hours clock.
- (3) The status of its two interrupt and mask registers.
- (4) Three indicators relating to self-test and power-code activity.
- (5) Output unit availability.

If the rollback feature is enabled (rollback refers to the capability of restarting a predesignated portion of a sequence), then its particular time/event region is flagged to be restarted if and when the PWRCHK routine requests it.

The PWRCHK routine is entered if 1) output unit initialization has occurred, 2) ERROR has successfully re-enabled itself and the power low-voltage response, and 3) the reason for entering ERROR was, in fact, a CCS tolerance detection trip or an undervoltage trip indication. Otherwise, the rollback table will be disabled and CCS will go to a WAIT state.

4.1 FUNCTIONAL DATA FLOW DIAGRAM

The functional data flow diagram is shown in Figures C8-2 and C8-3.

SECTION 5

SUBSYSTEM FUNCTIONAL REQUIREMENTS

5.1 INPUTS

The ERROR routine shall respond to the following hardware and software detected errors.

5.1.1 Hardware

- (1) A low voltage condition exists.
- (2) A primary command sync signal has been received before the previous one was processed.
- (3) An internal CCS error has occurred.

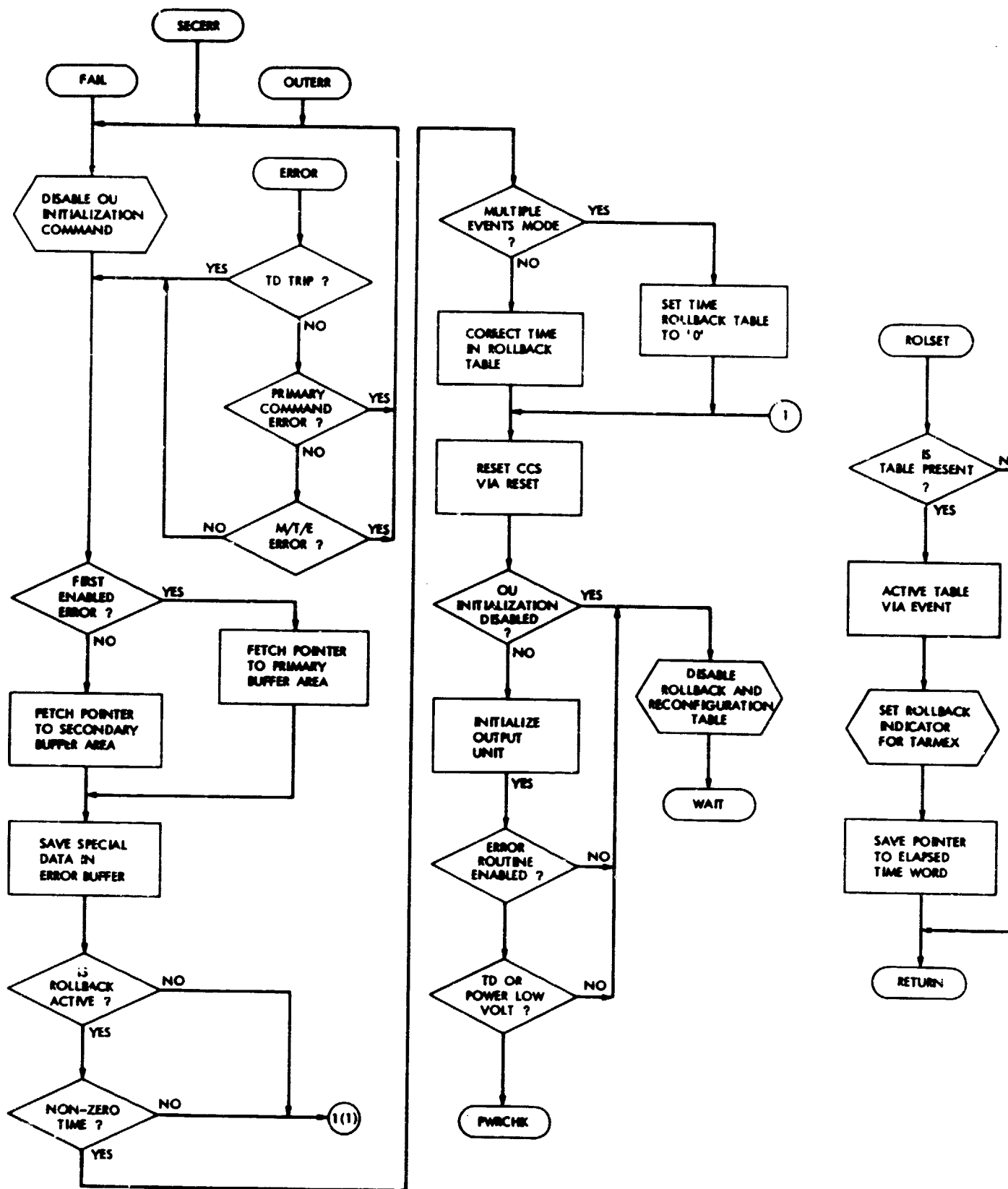


Figure C8-2. Flow Diagram, ERROR Routine

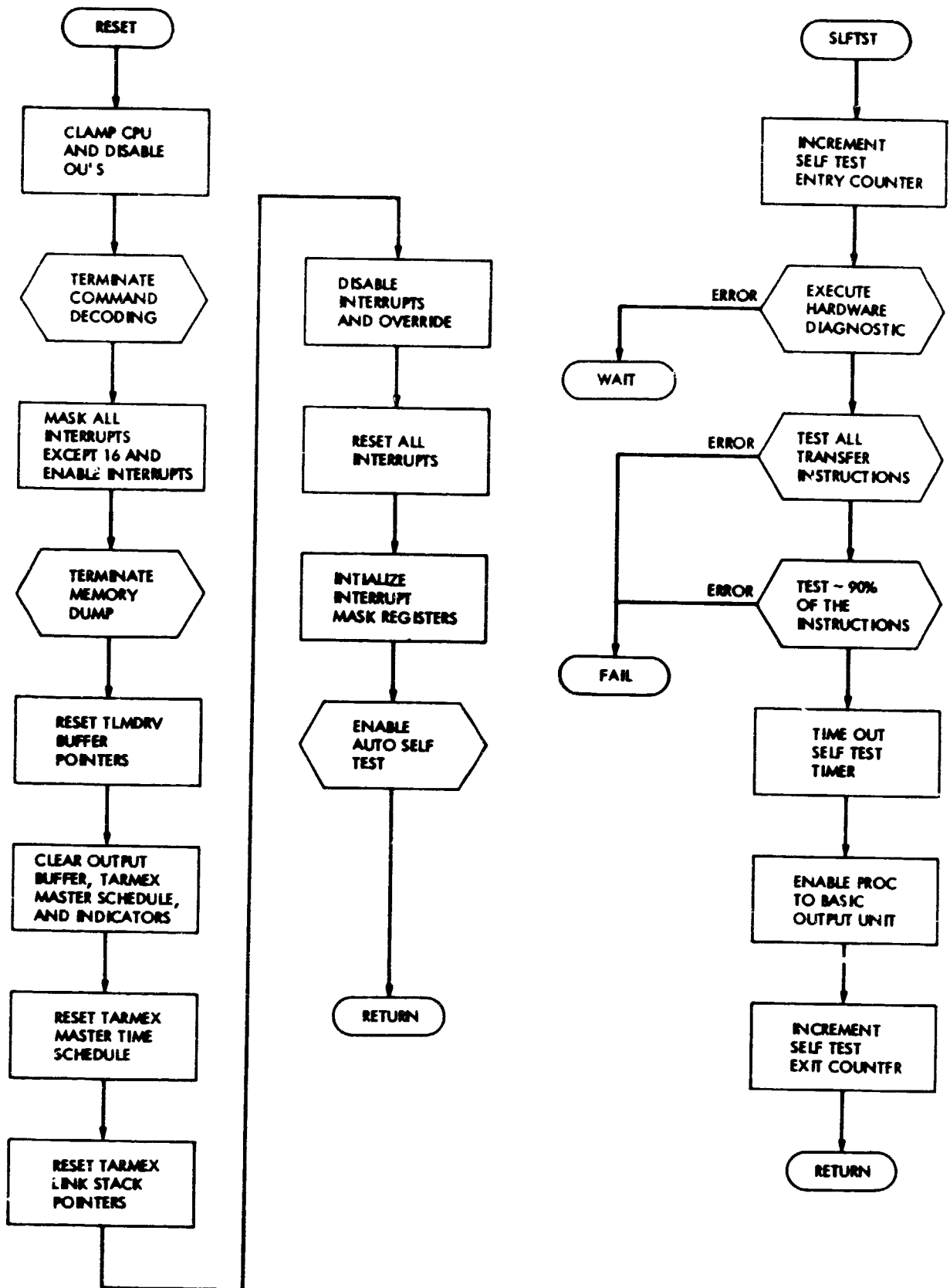


Figure C8-3. Flow Diagram, ERROR Routine

5.1.2 Software

- (1) The primary output unit has been unavailable for 14 seconds or longer.
- (2) The self-test subroutine has not executed properly.
- (3) A secondary command sync signal has been received before the previous one was processed.
- (4) The sequencing support routine (TARMEX) has been asked to activate more time/event tables than it can handle.
- (5) The output buffer has overflowed.
- (6) During launch a processor is counting fast relative to the other processor and the Flight Data System (FDS).

5.1.3 Rollback

The ERROR routine shall also respond to a rollback identification request from TARMEX. When this request is received, the ERROR routine shall set an indicator which shall identify the sequence to be restarted in the event of a power low-voltage condition.

5.2 PROCESSING

Immediately upon activation by any of the above anomaly conditions, the ERROR routine shall sample and store the condition of the following hardware/software functions.

- (1) Condition of hardware error flip-flops.
- (2) Status of interrupt register 1.
- (3) Status of interrupt register 2.
- (4) Absolute hour count.
- (5) Power Code active indicator.
- (6) Self-test active indicator.
- (7) Output unit exclusive-use indicator.
- (8) Status of mask register 1.
- (9) Status of mask register 2.

The routine shall provide the capability to store two sets of the above data for two consecutive error conditions.

The routine shall also check to determine if the rollback feature is enabled. If it is, the associated time/event table shall be flagged to be reactivated when the Power Recovery routine provides for it. A reset/initialization function shall be provided which does the following:

- (1) Ensure that the other processor is inhibited from communicating with its output unit, unless it successfully passes its self-test routine.
- (2) Terminates the following activities if in progress:
 - (a) Command decoding.
 - (b) Memory dump.
 - (c) Sequence activity (except rollback).
 - (d) FDS/AACS memory load.
 - (e) Power code processing (momentarily).
 - (f) DSS tape positioning.
- (3) Clears the following:
 - (a) Other-processor data.
 - (b) Output unit not-available time-counters (2).
 - (c) Output buffer.
 - (d) TARMEX time and block schedules.
 - (e) FDS/AACS memory load pointer.
 - (f) Power code processing indicators.
 - (g) DSS tape direction/active indicator.
 - (h) Power low-voltage enable.
- (4) Resets output and telemetry buffer pointers.
- (5) Disables interrupts and unmarks the following interrupts.
 - (a) Error.
 - (b) DSS tape inputs.
 - (c) Power codes for AACS.
 - (d) Internal interrupt (used during sequence execution)
 - (e) Checksum.
 - (f) Command decoding.
 - (g) Demand Read (data from other processor).
 - (h) 1 pulse per hour (PPH) timing input.
 - (i) Radio Frequency Subsystem (RFS) failure inputs.
 - (j) Power inverter switch and Infrared Interferometer Spectrometer (IRIS) supply failure.
 - (k) Self-test.

The output-inhibit clamp that each processor applies to the other shall always be in effect. The inhibit function shall only be momentarily overridden by successful execution of the self-test routine prior to the outputting of a command. In this manner, the outputting of an erroneous command by a malfunctioning processor shall be minimized.

After the reset/initialization function is completed, this routine shall initialize the output units for future use, provided that the detected error condition is not associated with a CCS processor-related error.

If the detected error is associated with a power low-voltage condition, then this routine shall transfer control to the PWRCHK routine, provided that such transfer has not been previously disabled by ground command, and that the prior output unit initialization has occurred.

If the detected error is not associated with power low-voltage, or the routine has been disabled from responding to such a condition, this routine shall disable the rollback function.

A self-test function shall be provided to verify proper operation of processor hardware and software prior to the output of a command to an external subsystem.

Only if the hardware/software test is completed successfully shall the processor be allowed to output commands to other subsystems. If an error occurs at any time during the test, the test shall be immediately terminated and no command enabling shall occur.

Specific actions taken in testing the hardware/software for correct operation shall be the following:

- (1) Set a test latch to indicate that self-test is in progress.
- (2) Set a test counter to zero.
- (3) Inhibit interrupts.
- (4) Turn off interrupts 2-8 enable override.

The test counter shall be used to control the hardware/software checks made during the self-test while the test latch is set. The test counter shall be set to zero when the test is initiated and incremented by one at Bit-Pulse-Last of every instruction unless the test counter has reached its final value of 127.

Test count cycles 1 to 15 shall be utilized to verify proper hardware register (memory address, memory data, program counter) operation. Test count cycles 16 to 127 shall be utilized to check software instruction execution. The instruction execution shall be designed to detect execution errors to the maximum extent possible. When an error is detected the test latch shall be reset as an indicator of erroneous operation.

The self-test shall be terminated unsuccessfully if any of the following conditions occur:

- (1) A constraint violation is detected by the hardware during test counter values 1-14 inclusive.
- (2) An error condition interrupt occurs.

The self-test shall be terminated successfully if the test latch is still set when the test counter reaches 127. In this case, command outputting shall be enabled.

5.3 OUTPUTS

The ERROR routine shall output an initialization command through the OUTDRV routine to the output units only if the error condition was due to a power low-voltage condition or a software error not associated with self-test. All other error conditions may interfere with the successful outputting of a command, and outputting shall therefore be inhibited.

If a sequence has been identified for rollback after a power low voltage condition, the ERROR routine shall set a special indicator for ultimate use by PWRCHK to re-enable that sequence.

An inhibit shall be applied to the other CCS processor during execution of this routine. The other processor may momentarily override this inhibit by successfully executing its self-test routine.

SECTION 6

INTERFACE LIST

A listing of the external interfaces is shown in Table C8-1.

SECTION 7

PERFORMANCE REQUIREMENTS

No additional requirements.

SECTION 8

IMPLEMENTATION CONSIDERATIONS

8.1 GENERAL CONSIDERATIONS AND RATIONALE

The ERROR routine was designed to terminate any pre-programmed sequence currently active, and to prevent any command output if the detected error condition might cause erroneous commands to be issued. This approach

Table C8-1. Interface List, ERROR Routine

FROM/TO	WHAT	HOW (MEDIUM)	INTERFACE DOCUMENT REFERENCE
PWR SUBSYSTEM/ERROR	POWER LOW VOLTAGE SIGNAL	HARDWARE INTERRUPT	PARA. 5.1; FIG. 2.1-1
CCS HARDWARE/ERROR	PRIMARY/SECONDARY COMMAND ERROR	HARDWARE LOGIC IN CCS	PARA. 5.1; FIG. 2.1-1
CCS HARDWARE/ERROR	INTERNAL CCS ERROR	HARDWARE LOGIC IN CCS	PARA. 5.1; FIG. 2.1-1
OUTDRV/ERROR	OUTPUT UNIT UNAVAILABLE	OUTPUT COMMUNICATION WITH OUTPUT UNIT	PARA. 5.1; FIG. 2.1-1
TARMEX/ERROR	TOO MANY SEQUENCE TABLES	TARMEX EXECUTION	PARA. 5.1; FIG. 2.1-1
OUTDRV/ERROR	OUTPUT BUFFER OVERFLOW	OUTDRV EXECUTION	PARA. 5.3; FIG. 2.1-1
ERROR/OUTDRV	OUTPUT UNIT INITIALIZATION	ERROR EXECUTION	PARA. 5.3; FIG. 2.1-1
ERROR/OTHER PROCESSOR	OUTPUT INHIBIT	ERROR EXECUTION	PARA. 5.3; FIG. 2.1-1

was dictated by the fact that command outputs out of each output unit (hardware) are wire-or'd. Therefore, any failure mode which could cause erroneous outputs from one half of the CCS interfering with the other processor's ability to output proper commands must take precedence over maintaining two active CCS halves. It should be noted that disabling a processor by its ERROR routine does not prevent receiving commands if there is no failure associated with that function. Indeed, commands would typically be used to read out failure indicators and status indicators following an error response. After the error was diagnosed and corrected, the processor would typically be re-enabled by ground command.

8.2 REQUIRED RESOURCES

The Voyager CCS ERROR routine requires 230 memory words in each CCS processor.

8.3 JPL EXPERIENCE

This routine has not been entered by any error condition on board the Voyager spacecraft since launch.

SECTION 9

VALIDATION AND TEST REQUIREMENTS

Validation and test of the ERROR routine shall be accomplished at two levels: subsystem and system.

9.1 SUBSYSTEM VALIDATION AND TEST

At the subsystem level, verification of the coded routine's ability to satisfy the specified requirements shall be accomplished by simulation of the execution of the routine on a test computer rather than the host computer. This shall allow easy variation of the input calls and data to validate proper operation of the routine.

As in any software system, individual routines or software modules are designed to operate in conjunction with other software routines or modules. Consequently, the ERROR routine shall utilize the following routines during simulation testing at the subsystem level:

9.1.1 Rollback Request

TARMEX shall provide the sequence-table rollback request.

9.1.2 Timing Inputs

TRAPS, COINTS shall provide timing inputs to TARMEX so that it can output the rollback request.

9.1.3 Output Unit Response and Commands

OUTDRV shall provide the output unit response error and process the output unit initialization commands from ERROR.

9.1.4 Power Low-Voltage Response

PWRCHK shall provide the response to the request from ERROR resulting from a power low-voltage condition.

9.1.5 Fixed Constants

GLBCNT shall provide the fixed constants used by ERROR during execution.

9.1.6 Variable Memory

VARABL shall provide the variable or 'scratchpad' memory required by ERROR.

9.1.7 Error Condition Testing

The hardware-related error conditions shall be tested by special simulator configurations during test sequences designed to test those functions.

9.2 SYSTEM (SPACECRAFT) LEVEL VALIDATION AND TEST

IBD.

AUTONOMOUS SPACECRAFT PROJECT

APPENDIX D

DATA PROCESSING ARCHITECTURES AND FAULT TOLERANCE SPECIFICATION

Appendix D

Section D1

A Centralized System Design Architecture for Autonomous Spacecraft Applications

Table of Contents

Appendix D

- SECTION D1: A Centralized System Design Architecture for Autonomous
Spacecraft Applications
- SECTION D2: A Decentralized System Design Architecture for Autonomous
Spacecraft Applications
- SECTION D3: A Recommended Hybrid Processing Architecture

INTRODUCTION

This report summarizes the results of a study to develop the benefits and the reasons for using centralized processing on autonomous spacecraft. A similar study developed the arguments for decentralized processing. For the sake of developing the arguments for and against either architecture, the definitions of the architectures were deliberately set at the extremes of the possible spectrum. Thus, neither architecture was necessarily intended to represent a likely candidate for flight use.

SECTION 1.0

CENTRALIZED SYSTEM DESIGN ARCHITECTURE FOR AUTONOMOUS SPACECRAFT APPLICATIONS

The architecture of a centralized system was defined to have the following characteristics:

- o All processing performed by one function.
- o Data from sensors supplied in raw or multiplexed form to the central processor.
- o Control commands passed to user functions and acted upon without further processing.

A simplified diagram of this architecture is shown in Figure D1-1.

Conceptually, all processing could be done by a single Central Processing Unit (CPU), however, multiprocessors were allowed for purposes of this study. Similarly, although conceptually a single Random Access Memory (RAM) and one mass memory could be used, multiple memories were permitted. In the remainder of this report "centralized processing" is used only to refer to the concept of performing all spacecraft processing in one place. In practice, a multiprocessor configuration would probably be used for better internal fault protection, flexible memory allocation, easing of processing speed constraints, etc. However, a specific multiprocessor architecture was not developed as part of this study.

SECTION 2.0

KEY CONCEPTS INVOLVED

The central processing function performs analysis of the state of all spacecraft functions and infers spacecraft status as necessary to ensure that the service functions, resource management functions and routine integrity maintenance (non-fault related) functions are being carried out as commanded.

Commands are generated by the central processor. In the absence of faults these are sequenced commands, or commands which are triggered by an expected condition (e.g., normal pointing errors, an eclipse, etc.). Commands are transmitted to the individual spacecraft functions which carry out the required actions.

Fault detection is performed by the central processor. Detection may be done from direct sensor inputs via a spacecraft function which indicates a specific fault, or by analysis of anomaly sensor data from one or more spacecraft functions. Faults are isolated to the necessary level by the central processor, and commands to switch components, change tolerances, etc., to correct the fault are issued to the affected functions. The central processor includes the spacecraft central executive. Executive functions include:

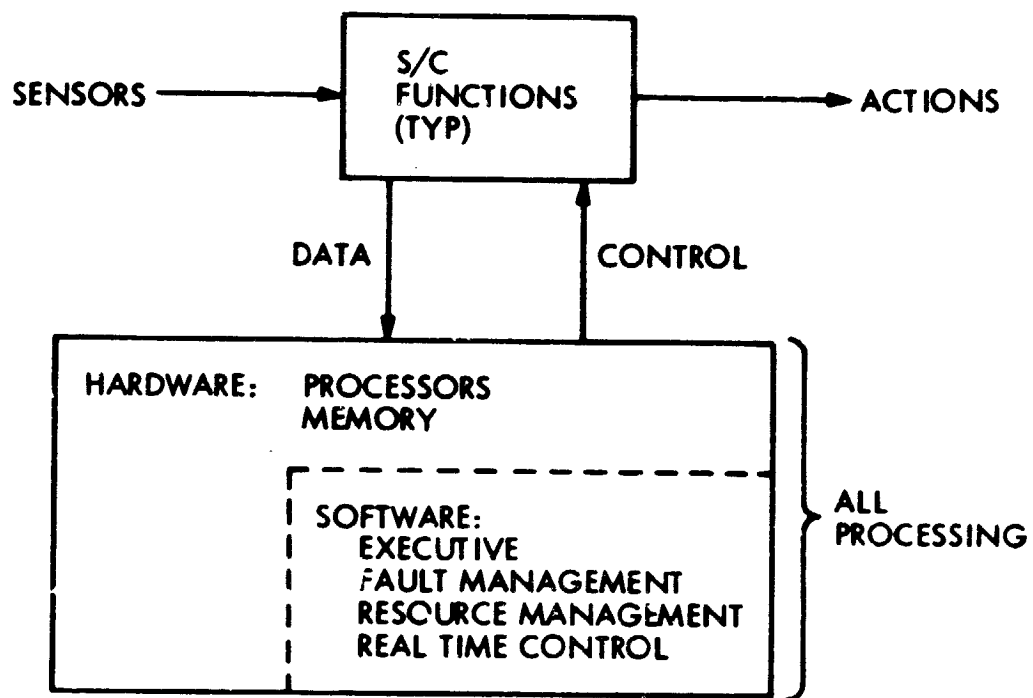


Figure D1-1. Centralized Processing System

1) system level decision making for allocation of critical resources, 2) priority arbitration in cases of competing subsystem requests, and 3) central coordination of sequences and fault responses.

More detailed architectural questions were not addressed. Unresolved issues include 1) the strategy to be used to protect the processor from internal faults, 2) the degree to which direct fault sensing would be used vs. inferences of faults through analysis, 3) the implementation of the executive functions, e.g., the degree to which a master/slave arrangement of the multiprocessor is utilized, and 4) assignments of processors to types of functions such as real time vs. batch mode processing.

Finally, architectural implications of payload data processing and payload control were not considered. Only spacecraft bus processing functions were addressed in the study.

SECTION 3.0

REQUIREMENTS THAT CAUSE SELECTION OF THIS ARCHITECTURE

A. Mission Requirements

All future Air Force satellite missions will have the following types of requirements for spacecraft bus functions which lead to a strong need for a central processing subsystem:

- o Normal operation without ground intervention.
- o High reliability.
- o Fault tolerance.
- o Long life with expendable resources.
- o Low risk implementation.
- o Minimal mass and power requirements.

The first four requirements imply that an executive control function is needed on the spacecraft to provide functional priorities and protocols, rapid decision making at the system level, system level resource allocation, and the central agency to analyze faults affecting more than one subsystem and to create the opportunity for functional redundancy.

Central processing systems are much better understood and therefore have lower risk than distributed processing systems. Mass and power requirements are probably inherently lower for central systems than for distributed because duplication of data bases and protocols is not required, and because a single internal fault tolerant system can be used instead of proliferating redundant processors throughout a distributed system.

B. System Requirements

1. Interaction Requirements

The interactive nature of the individual spacecraft functions is the primary driver toward central processing. Figure D1-2 is an "N²" matrix with the primary spacecraft functions along the diagonal. An X in the column containing a given function indicates that it requires an input from the function in the row that X is in. For example, the X in the box above attitude control indicates that power is required by the attitude control function. Similarly, an X in the row containing the function indicates an output from that function to the function in the column containing the X. For example, attitude control sends demands for power to the power subsystem.

Figure D1-2 illustrates clearly the highly interactive nature of the subsystems. This high level of interaction makes an executive control system imperative, especially for functions requiring prioritization such as resource allocation and fault correction.

2. Cycle Time Requirements

Other than payload, only one function is thought to require continuous high frequency data processing. This is attitude control. In the event of a fault, rapid response by the processor may be necessary, but fault protection activities will (hopefully) be infrequent. All other functions are not considered to be time critical. This means that attitude control will be the driver on the selection of the size of a computer processing cycle time. Enough additional space in the frame must be devoted to the routine functions which are not time critical but which must be monitored by the central processor. Fault protection events requiring real time attention by the processor can preempt the non-time critical routine events. Therefore, the CPU cycle time will probably be sized by the real time attitude control requirements plus the fault protection requirements.

3. Memory Requirements

In terms of active memory size autonomous navigation is likely to be dominant during periods when calculations are being performed. Estimates of active memory required for autonomous navigation may range as high as 32K, 32 bit words. Again, fault protection activities are likely to require large amounts of memory but should be infrequent. If mass data storage is available the fault protection routines could be stored and could preempt navigation and other non-critical routine activities when required. The impacts of this scheme should be minor, such as possible less efficient maneuvers or resource management. Therefore, active memory is likely to be sized by a combination of attitude control, navigation and fault protection, plus the executive functions to manage memory and cycle time resources. Other memory requirements are likely to be relatively minor.

EXEC	X	X	X	X	X	X	X
X	DATA STORAGE	X	X	X	X	X	X
	X	TELECOM	X	X			X
X	X	X	THERMAL	X	X	X	X
X	X	X	X	POWER	X	X	X
	X	X	X	X	A/C	X	X
	X	X	X	X	X	PROP	X
	X	X	X	X	X	X	NAV

Figure D1-2. Spacecraft Function Interactions

4. Data Base Requirements

The individual functions have considerable overlap in their needs for access to data, especially for fault protection. For example, the most likely way that a propulsion thruster leak will be detected is by its influence on the frequency of attitude control activities. Problems in the power loads are likely to be detected by a combination of thermal measurements and power measurements. Therefore, ready access to a central data system by the fault protection function is essential for reliable fault detection.

5. Control Requirements

There is also considerable overlap in the requirements for functional control of the spacecraft. For example, thruster firings are required for both attitude control and navigation. Switching of redundant elements in all functions is done through the power function. Because such overlaps will result in priorities and protocols being required to control the spacecraft, control is most efficiently invoked by a central processor.

6. Mass/Power Requirements

All Air Force missions have severe constraints on spacecraft mass and power. This means that duplication of computers must be kept to a minimum. Since processors will probably be made fault tolerant through provision of redundant processors for internal fault checking, a central processor subsystem will require fewer redundant computers than a distributed processing system, where each processor must have two or more redundant backups.

A central processing subsystem can also make the most efficient use of computer resources through management of its multiple processors.

7. Summary of Spacecraft Requirements

Requirements which drive the autonomous spacecraft toward a central processing subsystem include:

- a) The highly interactive nature of the functions requiring a central control and data base.
- b) The fact that only one or two functions drive cycle time and active memory size requirements. Thus, dedicated processors sized for fault protection of specific functions would generally be "idling" during normal operations.
- c) The extreme likelihood that fault protection (at least) will require an overview of fault symptoms from several functions, and executive control to correct faults involving or affecting more than one function.
- d) The need to eliminate excessive duplication of computer resources from a mass/power limitation standpoint.

C. Project Requirements

Air Force satellite missions have strong requirements for long-life/high reliability hardware and software, and for low risk. Central processing systems, because of the experience base of industry and the Air Force, are much lower risk than distributed processing systems. NASA's Galileo spacecraft will have the most distributed processing system ever flown and this system features control by a central processing subsystem.

Thus, the Air Force requirements for low risk drive processing strongly to a centralized system.

SECTION 4.0

BENEFITS AND DETRACTORS

The benefits of centralized processing are mainly advantages in system design and implementation, many stemming from the use of a central executive. The detractors are the magnitude of the job of managing one dominant subsystem and the design and test constraints imposed on other subsystems.

A. Benefits

Advantages to centralizing the processing function fall in the general areas of:

- o Benefits of a central executive.
- o Simplifying system test and validation.
- o Design and control inherent in a single management area.
- o Simplified control of interactive functions.
- o Potential system mass, power and interface advantages.
- o Accumulated industry experience in management and technical approaches.

1. Central Executive

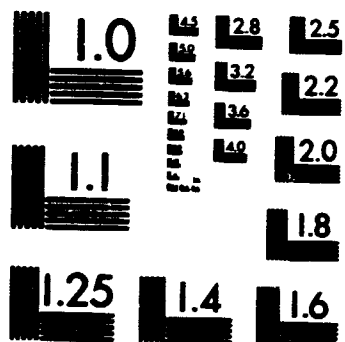
A centralized executive function will almost certainly be required in a Level 5 autonomous spacecraft design and is currently used in planetary spacecraft designs, including Galileo, with lesser degrees of autonomy. The executive provides the system level decision making capability for situations where subsystem decisions are not permitted or must be influenced by information not normally available to the subsystem. Examples include:

- o Critical resource management - Control of both computer resource (processing time, memory space) and expendables (propellant, power, spares) must be centrally maintained to ensure system response capability in the face of competing demands by subsystems.
- o Priority arbitration - Design of sequences for normal spacecraft operation or during fault correction attempts must properly reflect mission and system requirements (rules) and may require delaying or ignoring non-critical subsystem actions.

83

5920

3/B



- o Response interpretation - Faults may produce symptoms in more than one subsystem which can result in multiple (perhaps conflicting or improper) responses. Central diagnosis and/or response is required to avoid these problems.

The central executive has access to all data, since it is by definition a part of the central and only processing function. System processing overhead is reduced compared to a completely distributed executive where it would be necessary to duplicate the executive in each of several processors.

A centralized executive should also be more adaptable to including functional redundancy for key spacecraft functions. Functionally redundant implementations often increase subsystem-to-subsystem interactions and produce different responses during faults, characteristics which require more centralized control.

2. Test and Validation

Centralized processing provides advantages in testing and validating the spacecraft and its software at the system level. However, complete centralization may hamper development and test of some subsystems prior to integration at the system test level. The following addresses only the spacecraft system level test and validation.

- o Response verification - Since a single system verifies response to stimuli or faults, improper responses are easier to trace and trouble-shoot.
- o Time correlation/time compression - Correlating time and determining the actual sequence of events (e.g., during fault detection/correction actions) is simplified when processing and control are handled by a single machine. (This attribute of centralized processing applies to in-flight operation as well as ground test). Time compression to accelerate or step through a test sequence can also be readily accommodated.
- o Memory access - Centralizing all memory results in simplifying processes requiring dumping the memory and verifying contents. Correlating and trouble-shooting distributed memory systems which are supporting interactive processing functions can be difficult.
- o Audit trail - Reconstruction of fault activities, when performed by a centralized processor and memory, results in a clean, time correlated record of events which can be readily compressed for storage and later retrieval.
- o Validation tools - Techniques and procedures for testing and validating central processing systems are well developed and in use by industry.

- o Simulation - System level simulations are simplified since the numerous parallel paths possible in a distributed system are not required. The ease of simulation of central processor systems also carries over from the spacecraft system test phase to mission operations including operator training.

3. Single Management Agency

Inherent in the central processing approach is a single management agency for procurement and development of hardware and software. Advantages in hardware development and procurement are obvious and may be shared by distributed approaches depending on the implementation mode. Software management advantages with centralized processing can be significant and fall into two broad areas:

- o Technical consistency - Development and validation of all software under close control of a common systems management entity reduces the chance for differing application or interpretation of system requirements. Specifically, close control of the following is required:
 - interfaces
 - priorities
 - timing
 - response interpretation
 - conventions
 - language
- o Programmatic control - Control of all software development by a single management agency provides better cost and schedule control and simplifies control of the software system design and change control during development. With software management under the control of one agency, problems are easier to identify and manpower allocations can be adjusted to suit. A team approach, with all involved areas represented, can be implemented easily.

4. Interactive Functions

As discussed earlier, Air Force spacecraft requiring autonomous control will consist of highly interactive subsystems. Centralized processing provides significant advantages over distributed processing when the functions being performed must be closely coordinated, as will be the case on future spacecraft (see Section III, B, 1). Advantages can be summarized as follows:

- o Data availability - All subsystem and system data required for real-time spacecraft operation or for fault protection analysis are available in a single memory. Recall of required data is limited only by internal processing priorities which can be internally modified as required.
- o Event coordination - Central control of real-time operations and fault management ensures that commands issued to subsystems are sequenced such that conflicting or improper spacecraft responses are prevented.
- o System response - The potential for rapid data recall and rapid restructuring of processing priorities ensures that rapid spacecraft response to events can be provided if required. Reaction time can be tailored to the need by temporarily suspending low priority processing.
- o Complex faults - Many faults involve more than one function. Fault analysis often requires data from several subsystems and isolation and correction may require that several subsystems respond. Central processors are well suited to handling these conditions. All data, fault processing and sequencing of responses are under central control ensuring that fault diagnosis examines all symptoms, regardless of where they appear in the system. Responses must correct all fault conditions, again regardless of where they appear, while observing system and subsystem constraints. Central control of these multiple, overlapping, complex actions would appear to be mandatory.

5. System

Centralizing the processor and memory has potential mass, power and volume advantages over distributed systems since duplication of functions is reduced or eliminated. Interfaces between subsystems are greatly simplified, being limited to subsystem to computer interfaces of little complexity such as relays, analog signals or coded commands.

A centralized processor is less susceptible to Electromagnetic Interference and the potential for data errors since extensive data bus networks are not used. Radiation shielding may be more easily applied to a centralized system (if hardened parts are not available) since susceptible parts will tend to be located in only a few chassis.

6. Industry Experience

Spacecraft experience to date has been with highly centralized systems. Technical and management approaches have been developed, certainly not to perfection, but to the point of coping with large dynamic systems such as the Shuttle Orbiter. Galileo, while characterized as a distributed system, retains a central executive and draws heavily on past JPL management and technical experience with relatively centralized systems.

While past experience is not, per se, a reason to continue doing the same thing, radical departures from what is known to be a workable approach must be carefully planned and the increased risk to Project success well understood.

B. Detractors

Potential problem areas include the following:

- o Accommodating multiple, high demand users
- o Processor sophistication required
- o Magnitude of the job
- o Subsystem design and test constraints

1. High Demand Users

As spacecraft complexity increases, the potential for multiple high rate users competing for limited CPU time will increase. Typically, the attitude control subsystem places the highest demands on the processor, both in terms of frequency and volume of computations required.

Payload data processing was not considered in this study, consistent with the current ASP approach. However, if a single central processor truly was a viable approach to autonomous spacecraft needs, payload data processing requirements, if any, would have to be included. Processing loads will depend on payload and mission, but could be significant.

Another potentially large user is fault protection. A centralized processor may be required to handle a large volume of routine computation as a continuous background function. However, since faults occur infrequently, fault correction actions should not often interfere with other real time spacecraft processing. Note that the volume of routine computation will depend on the success in detecting and isolating spacecraft faults directly (e.g., using sensors). If most faults must be inferred from analysis of spacecraft reaction, the computing volume can be quite large.

To the extent that payload, fault protection or other large users compete with attitude control for processing time, a single central processor may be unable to handle all demands and cycle slips, etc., may result. A multi-processor architecture may be required to handle large demands.

2. Processor Sophistication

Because of the large volume of processing to be handled by a single central processor, a relatively powerful machine is required. This tends to limit the number of choices available to the spacecraft designer. Requirements for radiation hardening further limit the available devices.

Within limits, this problem may be eased by multi-processor architectures.

3. Magnitude of Job

The job of managing a large, centralized processing subsystem could easily dominate the autonomous spacecraft design and management structure:

- o The software design and integration process is complex and highly interactive, requiring strong teams of subsystem and system experts during design, development, test and flight phases.
- o Reprogramming may become difficult late in the development phases unless reserves in processing time, memory and program structure are carefully protected.
- o Placing all processing resources in one place increases the potential for the processing function to become the overall project schedule driver at any time due to either hardware or software problems.

4. Subsystem Design and Test Constraints

While centralizing the processing function has potential advantages to the system design and test phases, it may pose problems to some subsystems during their development and test programs. Subsystems which are highly interactive with software, such as attitude control, will require system software simulators which must be maintained and updated by the central subsystem agency. Such subsystems may find themselves in series with the software developers and subsystem delays may result. Careful planning and coordination will be required to minimize these and other problems which stem from subsystem dependence on timely and successful software development by a separate agency.

SECTION 5.0

PERFORMANCE FEATURES

A. Executive Control

An adequate executive gives the capability of providing all aspects of control required for the efficient functioning of a computer. The control is centralized so that it will eliminate conflicts which may arise due to competition for resources by the subsystems which interface with the computer.

The control is over the following:

- o Competition for bus accessibility
- o Competition for memory resources
- o Dynamic or fixed resource priority
- o Assignment of memory

- o Assignment of diagnostic modules
- o Substitution of failed modules

B. Simplicity of Subsystem Interaction

There is a minimum need for complex interface logic or hardware. Programming provides all of the linkage required between subsystems. This eliminates the need for complex protocol or interaction between the subsystems.

C. Reduced Overhead

Overhead may be defined as non-productive use of resources. The inefficient use of time, memory and program space may be considered as contributing to higher overhead.

The exchange of information within a centralized system is facilitated by its executive. The integration of programming affecting two or more subsystems reduces the amount of time for processing information as well as the possibility of the need for making several exchanges of information between subsystems. Memory sharing, under executive control, is another advantage of a single processor/memory. Any number of functions may time share a common memory, thereby reducing the need for dedicated memory for each subsystem. The use of subroutines which can be shared by several subsystems will reduce the amount of program space required.

D. Elimination of Bus Contention

Bus contention can be a serious problem when several users of a computer bus require its services. The use of protocol becomes very cumbersome in the absence of any central control. The need for protocol is eliminated by the action of the executive, which has advance information or develops the information which chooses the bus user.

E. Multiplexing and Demultiplexing

A single analog multiplexer with inputs from many sources, under executive control, may be used to address a single analog-to-digital converter. The digital result, again under executive control, can be put onto the computer data bus for processing the data. Demultiplexing to the appropriate subsystem memory is inherent in this control.

F. Using Special Function Modules

Many modules are available for performing logic, making decisions and doing mathematical operations. They include devices to perform:

- o Fast multiplication and division
- o Trigonometric calculations
- o Code conversion
- o Frequency component determination by fast Fourier transform (FFT)

Such modules are individually addressable and can reduce the burden of doing complex calculations in a very efficient manner. A centralized system has the advantage of being able to share these functions with many subsystems.

G. Fault Detection, Diagnosis and Correction

These are separate functions which are closely linked in a centralized system. Each of these functions may be further modularized into subroutines which may be chosen as appropriate under executive control. The subroutines will probably be shared by a number of subsystems. Faults which cross subsystem boundaries are much easier to detect, diagnose and rectify in a centralized system.

H. Intervention Under Stress Conditions

Stress conditions will exist with the sudden appearance of an emergency condition. This may involve the need for rapid interaction between two or more subsystems, and may require an interrupt capability. The timely management of stress processing is well suited to executive control.

SECTION 6.0

FLEXIBILITY

Maintaining flexibility in a centralized processing system requires some degree of effort and preparation. The preparation for flexibility requires the recognition that extra memory is required to provide additional programming space and data repository. The insertion of a small program within the body of a large program, although not difficult in itself, may have implications for the entire program sequence.

The memory required for additional programs and data must be anticipated and provided. Ease of program insertion must be provided. If there is not enough random access memory available for the total program and data storage, it may become necessary to use mass memory as a source for infrequently used programs and as a repository for extra data. This may reduce the speed of a centralized system, but if intelligently used, will increase its capability with small sacrifice.

SECTION 7.0

CONSTRAINTS

A series of characteristics of a centralized system can be considered to constrain the system design. The following issues also serve to point to possible design trade-offs that would ease the constraints or allow the best resolution of them.

A. Executive Software Complexity

The central executive software module is responsible for all subsystem resource management, control over execution of software modules for other subsystems functions, and input/output interfaces. This, plus the additional requirements of supporting fault tolerant modes of operation leads to a rather complex set of functional requirements. The development of executive software that meets all those requirements without usurping a major portion of system resources itself is a major design constraint.

B. Resource Margins

The flexibility and reprogramability of a centralized system is directly related to the amount of unallocated resources remaining after a baseline set of functions has been implemented. Computer and memory sizing are normally performed fairly early in the design process, and it is difficult to estimate margin requirements until some experience is gained with a candidate design. Memory size is the most visible component of resource for margin management, but input/output capability and execution speed and cycle timing can also lead to later design conflicts between hardware and software.

C. Resource Allocation

The allocation of common resources to functions that must execute under real time control is a major constraint of a centralized system. This process is reasonably well understood, and methodologies have been developed for dealing with the problem successfully. A complex, fault tolerant system still is a significant design problem.

D. Centralized Risk Factor

The centralization of processing also makes the system a schedule driver for the entire spacecraft. Any hardware faults, parts availability or unresolved design problems will impact the entire effort. Subsystems may not be able to carry on their own development process without the service provided by the centralized computing system.

E. Internal/External Fault Identification

The centralized computer has logical control of fault management. Upon receiving symptoms of a fault, it must be able to differentiate between occurrence of the fault in an external subsystem, a communications link with the external subsystem, or within its own software or hardware. This may be difficult without careful design of the fault indicators in external subsystems and the self test design of the central computer hardware/software.

SECTION 8.0

KEY DESIGN TRADE-OFFS

A. Coded Logic vs. Protocol and Standards

The amount and complexity of requirements on the executive software leads to a need to provide for satisfaction of requirements without usurping the entire machine resources. Coded logic can provide for flexibility and explicit treatment of selected functions. Carefully selected interface or logical design standards/protocols may relieve the executive code overhead by providing for a requirement in its entirety or by relieving the amount of coded logic needed to implement the requirement explicitly.

B. Function Execution Scheduling

Several design techniques are available for scheduling and controlling real time resource allocation. Trade-offs to select a specific method should take into account the basic characteristics of the external subsystem processing requirements, intercommunications requirements and constraints of a specific hardware architecture. Some specific techniques available for consideration are sequential execution with a priority interrupt for contingencies, preemptive scheduling where time "slices" are allowed for partial execution of individual functions, or some hybrid blend of these techniques.

C. Uniprocessor vs. Multiprocessor Architecture

Real time resource contention may be alleviated by providing several processors under control of a centralized executive processor. This decision has a great impact upon overall system design, and must be decided fairly early in the system design process.

D. Communication Protocol

Data from external subsystems could be processed by an interrupt to the main executive or a temporary storage protocol could be adapted to allow the data to be located when software functional processing requires it. An additional detail would be whether a function issues a command to an external subsystem for data and waits for a reply, or if data is collected as available from the subsystem.

E. Fault Instrumentation vs. Inference

Fault detection can be implemented by instrumenting a subsystem to detect specific identifiable fault conditions, or the existence of a fault may be inferred from health measurements, heartbeat checks, etc. Each technique is appropriate for some types of fault but not necessarily for all. Trade-offs need to be made on the basis of characteristics of identified faults and the planned availability of health data and its contribution to an unambiguous fault indication.

F. Fault Tolerant Computer Architecture

A choice must be made of a central processor of a "fault tolerant" design (i.e., multiple processors in parallel voting on results) or a multi-processor cluster with a fault monitor which can redistribute the processing load upon detection of a fault. This choice will also affect the relative role of hardware and software in the fault detection process.

SECTION 9.0

OTHER DRIVERS

Several factors emerged during this study which could influence architecture selection but which could not be addressed in the time available.

A. Payload Data Processing

As discussed, this study did not include payload data processing requirements. Since the payload could become one of the dominant users of processing resources, it may be influential in determining computer architecture. Future studies should reassess the current separation between bus and payload functions.

B. Sensed vs. Inferred Fault Detection

The processing load will increase as faults are determined by inference from "indirect" indications requiring some form of onboard analysis. The processing load will decrease as faults are directly and unambiguously sensed at the source. The amount of inferential analysis which will be required to achieve the ASP goals of Level 5 autonomy/six month autonomous operation should be assessed.

C. Central Processor Architecture

Multiprocessor architectures which preserve many of the advantages of centralization but overcome the problems in coping with ever-increasing processing loads are possible. Development of specific centralized multiprocessor architectures for autonomous spacecraft should be included if additional studies of this type are attempted.

D. Computer Fault Protection

Fault protection for a central computer can be provided in several ways. For this study it was assumed that multiple, parallel processor/memories could be used. More efficient schemes including internal hardware and software self-checking were not examined. Much work has been done in this field and should be included in future architecture studies.

Appendix D

Section D2

A DECENTRALIZED SYSTEM DESIGN ARCHITECTURE
FOR AUTONOMOUS SPACECRAFT APPLICATIONS

I. INTRODUCTION

The early use of computer technology for spacecraft applications favored a centralized processing architecture based upon both spacecraft needs and implementation practicality. For these early applications, only a limited few spacecraft subsystems required computer support to perform their service functions. Furthermore, early spacecraft computer designs placed significant demands on spacecraft mass and power. An example is the NASA standard MMS spacecraft using the NSSC-1 central computer. The centralized CPU/memory capability housed in the Communications and Data Handling (C&DH) module is almost fully utilized by the needs of Attitude Control. Mission applications having other high-demand users, such as a sophisticated payload, would necessitate either 1) a more powerful centralized computer or 2) the use of multiple computers.

There is a practical limitation to how powerful one can make a single computer in terms of throughput rate. This limitation coupled with the emergence of microprocessor technology has resulted in a recent trend towards distributed processing architectures using multiple computers. An example of the use of such an architecture is found in the JPL Galileo spacecraft currently under development. The Command and Data Subsystem (CDS) functions as a central executive computer that provides high-level control via a supervisory bus to distributed microprocessors. This allows the central computer to be offloaded so that increasing demands on workload and throughput, resulting from more sophisticated spacecraft needs, can be practically accommodated. The Galileo decision to distribute some of its processing was primarily driven by the instrument computing needs of its relatively sophisticated payload.

As the complexity and sophistication of future spacecraft continue to grow, the trend should continue towards increased distribution of the processing function to the subsystem level. This could be extended to a fully decentralized system design architecture consisting of discrete and relatively system-independent subsystems capable of being tested and validated outside of the system. Such practicality should allow potentially large reductions in system operational interface complexity, and would translate to significant cost reductions for spacecraft integration, test, and operation.

The following two factors should tend to drive future spacecraft designs toward fully decentralized architectures:

1. The need for computing support by every spacecraft subsystem.
2. The availability of reliable, cost-effective, self-checking, fault tolerant computer modules having low weight and power characteristics.

The first item should be satisfied by the future need for autonomy. The requirement of even the most simplistic subsystem to be autonomous with respect to integrity maintenance implies the application of computer software support to fully meet the diagnostic and recovery needs associated with fault detection and correction. The second item should be satisfied in the near future by technology advancements in the areas of microprocessor, memory, and Very Large Scale Integration (VLSI) development. In fact, JPL is currently breadboarding a self-checking, fault tolerant computer that uses VLSI-compatible building block modules to interface with commercially available microprocessor and memory chips.

The following sections describe and evaluate a possible candidate architecture for a fully decentralized spacecraft processing system. In Section II, the system design architecture is described and assessed. In Sections III through IX, the individual subsystems of the system defined in Section II are described and evaluated. For consistency, Sections II through IX are each organized to the following format:

A. DESCRIPTION

1. Function
2. Functional Elements
3. Functional Requirements
4. Interface Requirements
5. Block Diagram

B. EVALUATION

1. Benefits
2. Performance Features
3. Flexibility
4. Constraints
5. Tradeoffs
6. Other Drivers
7. Detractors
8. Conclusions

Finally, Section X addresses test and validation methodology associated with the foregoing system and subsystem design architectures.

II. SYSTEM DESIGN ARCHITECTURE

A. DESCRIPTION

1. Function

The function of the decentralized system design architecture is to fully perform spacecraft (and payload) useful services, resource management, and integrity maintenance commensurate with the needs of a designated mission through allocation of all required computer processing to the subsystem level with no centralized system level control.

2. Functional Elements

The decentralized system design architecture consists of seven computerized fault tolerant subsystems. They are defined as follows:

a) Common Memory Subsystem (CMS)

A description of the CMS is provided in Section III.

b) Telemetry, Tracking, and Command Subsystem (TT&C)

A description of the TT&C is provided in Section IV.

c) Payload Subsystem (P/L)

A description of the P/L is provided in Section V.

d) Navigation Subsystem (NAV)

A description of the NAV is provided in Section VI.

e) Attitude, Translation, and Pointing Subsystem (ATPS)

A description of the ATPS is provided in Section VII.

f) Temperature Control Subsystem (TCS)

A description of the TCS is provided in Section VIII.

g) Power Subsystem (PWR)

A description of the PWR is provided in Section IX.

3. Functional Requirements

Selection of a spacecraft computer system architecture is influenced by many factors. Only a few of these are a) the familiarity of designers and testers with presently used systems, b) data transfer speeds, c) complexity of fault routines, d) redundancy levels, e) computer conflict probabilities, f) mass and power efficiency, g) induced noise levels and suppression techniques, h) interface circuit complexities, i) overall ease of validation, and j) multi-mission applicability. One of the choices available to the spacecraft designer is that of a decentralized processing structure whereby computing capability resides in the subsystems, and decisions are controlled at this level rather than by a spacecraft central executive. For application to a generic spacecraft, one may formulate sets of system and subsystem functional requirements which specifically concern this architectural choice, but which preclude neither a mission of any character nor autonomy at any level. These requirements may be further characterized by mission phase, i.e., starting out with buildup and checkout, then moving into operations. A candidate set of such system-level functional requirements applicable to a decentralized system design architecture is defined as follows:

a) Initialization and Checkout

There are numerous reasons for launching the spacecraft in a quiescent state, not the least of which is that the spacecraft undergoes conditions during launch which would falsely indicate gross failures if its operational sensing devices were on-line. Therefore the spacecraft shall be launched in an "off" state with only essential loads on line.

Secondly, the on-orbit initialization actions shall be ground commanded. This implies that all autonomy routines shall be disabled through the launch phase and brought on line when the "mission" phase begins. Autonomy itself shall require subsystem internal power switching for redundancy control.

Thirdly, prelaunch validation shall be complete to such an extent that little or no on-orbit checkout shall be necessary, with the exception of sensor calibrations. Onboard actions, especially within fault routines, may be so disguised that test equipment will be necessary to trace signal paths and monitor actions/responses. Given that probability, the necessary orbital checkout shall be limited to only those verifications associated with the initial ground commands.

b) Ground Interactions

From the initialization period on, contact with the ground shall be considerably less than that realized in current practice.

However, the decentralized system shall be receptive to ground contact at all times. Even though complete operational and fault recovery logic shall be resident on board, sequences may be changed or updated periodically depending upon mission phase or changes in requirements desired by the ground. Furthermore, all internal self-test type actions shall be made visible to the ground upon request. Inquiries may be transmitted periodically to check on, for example, anomalous telemetry information or a faulty heartbeat signal. Self-checking procedures shall normally be accomplished without the benefit of ground intervention or visibility; however, as above, the availability of these functions for monitoring shall be present.

c) Mission Related

The payload on the candidate spacecraft shall be considered as an independent subsystem subject to the same requirements and constraints as all other subsystems. Any subsystem interactions shall occur through the normal media for spacecraft communications and shall not impose special information processing requirements on the remainder of the spacecraft.

d) Operations

Along with any set of advantages for a concept comes a set of commensurate disadvantages. One attempts to circumvent these conditions through additional system requirements. The very nature of the decentralized subsystems and their independence of operation inherently does not foster feelings of comfort in mission operations personnel. They are comforted only by complete, up-to-date knowledge of spacecraft status. There are several requirements which have been generated because of the independent nature of the subsystems in a decentralized system. These are discussed as follows:

It is possible for a subsystem to be malfunctioning in a passive sense, yet not be identified as such until some later time, if the subsystem has little contact with others on a routine basis. In order to insure proper functioning, each subsystem shall furnish a heartbeat type indication to the normal telemetry stream and to the audit trail. Routines shall be written so that the presence of telemetry itself does not furnish the same indication as does the heartbeat code.

Since it is also possible for two or more subsystems to be generating near-identical information in their independent implementations, such information shall be validated on board before usage. One such method could be a voting logic scheme.

It should be emphasized that precluding subsystems from duplicating sensor information is not necessarily the intention. For instance, there may be limitations on an information transfer, through the media provided by a decentralized architecture, which are unacceptable to a certain subsystem. The validations are required to insure that systematic error buildups do not cause related subsystems to become non-synchronous with regard to their basic parameters.

Another consideration involves error conditions in one subsystem caused by non-error conditions in another subsystem. Because of subsystem independence, a control subsystem like ATPS has the capability to generate an error symptom in a sensing subsystem like PWR when, for example, the earth presence signal is lost. Whenever a similar occasion is prompted, any subsystem which generates this unique and temporary condition shall notify other affected subsystems to disable their sensing or switch fault routines for a specified time interval contained in the message. These false error conditions can be well defined once the mission is known and can be updated when necessary as mission experience is gained.

Normal mission operations, irrespective of the independent aspect of subsystems, leads to three additional requirements. These are discussed as follows:

There is a need to keep a continuous record of mass and inertial property changes on-board for use in the ATPS maneuver calculations. Subsequent to the spacecraft initialization, the ATPS shall monitor propellant usage and configuration changes; then, it shall calculate and retain a record of the effects of these disturbances for trend analysis and real-time processing needs in any on-board subsystem. All of the raw information and/or the final calculations may or may not be furnished external to the ATPS depending upon the ascertained needs of other subsystems.

Audit trail and status reporting information shall be stored in a hardened non-volatile area of a common spacecraft memory to fulfill the overall 6-month performance analysis requirement currently defined for autonomous spacecraft. This shall be done in lieu of partitioning that storage responsibility among the individual subsystems. Appropriate formats shall be defined during the system design phase.

Finally, subsystem-internal power switching for most subsystem components shall be a necessity. This implies that after the initial orbital power-up phase, which will be largely PWR controlled, component control in a decentralized architecture shall reside within the individual subsystems to the maximum extent possible. Therefore, PWR must either guarantee power quality and presence, or tradeoffs involving non-volatile memory and/or energy storage methods will be necessary to insure that routines are not interrupted, or that critical memory contents are not lost.

4. Interface Requirements

It will not be possible to carry the subsystem independence condition to its literal limit. There remains a subset of requirements on the interfaces between the individual subsystems, which at this time can only be listed as a partial summary. Interfaces are typically a very difficult area to bound. A complete set of interface definitions can only result from comprehensive reviews during the systems definition and design phase of a flight program. One can only start an initial listing here. However, the following interface requirements for a decentralized system design architecture stand out immediately.

Power shall emanate from a central source. A dc-voltage level shall be generated within the power subsystem and supplied, unregulated, to each user. Redundancy details and methods for conversion and regulation shall be determined as part of the subsystem designs.

Timing shall also be provided from a central location for synchronization of data transfer, event execution, and sequencing in general. Its source may be one of several possible subsystems depending upon the results of a system design tradeoff study. The timing signal shall enable all on-board communications.

Access to a common spacecraft memory shall be each subsystem's vehicle for external communication. In order to partition this access, each subsystem shall be allocated a cyclic time slot for either read or write usage of the memory. This time slot shall be typically on the order of 100 ms and the cycle time shall be on the order of one second.

An implementation of the above capability will require processing within each subsystem. Therefore, each subsystem of the decentralized spacecraft architecture shall provide for a micro-processor, resident memory, and the appropriate I/O ports necessary for the communicative and processing functions referred to above.

The power subsystem shall post maximum power limits for each of the operating subsystems during each mission phase. With a maximum power availability limit for the spacecraft as a whole, each subsystem cannot have complete freedom of operation, or demand could exceed supply. The power subsystem shall exercise control over the rationing.

The last requirement deals with the natural flow of information to the common spacecraft memory and its content. Although every consideration should be given to generating needed information within any one subsystem, it may be more to the system's advantage if it were generated elsewhere. When this is the case, the selected subsystem shall furnish the needed information to the memory for general availability. This category shall be kept to a minimum to keep independence maximized and traffic external to the subsystems at a minimum.

5. Block Diagram

A functional block diagram for the subject system design architecture is given in Figure D2-1. Referring to Figure D2-1, each of the seven subsystems defined in Section II.A.2 autonomously performs its designated service functions and maintains its own health and welfare with no executive control from external spacecraft sources. The only executive control is provided via the spacecraft RF command channel from mission controllers on the ground.

A key concept associated with the architectural design of Figure D2-1 is that the Common Memory Subsystem (CMS) represents the only information transfer interface for all other spacecraft subsystems. Each subsystem (including the CMS) provides a priori defined internally generated digital information via the Intercom bus to appropriate addresses in the CMS memory for access by other subsystems. Externally generated information required by a subsystem to perform its function can then be accessed by the subsystem through periodic interrogation of appropriate CMS memory addresses. This information may be in the form of spacecraft parametric data from other subsystems, prioritized requests from

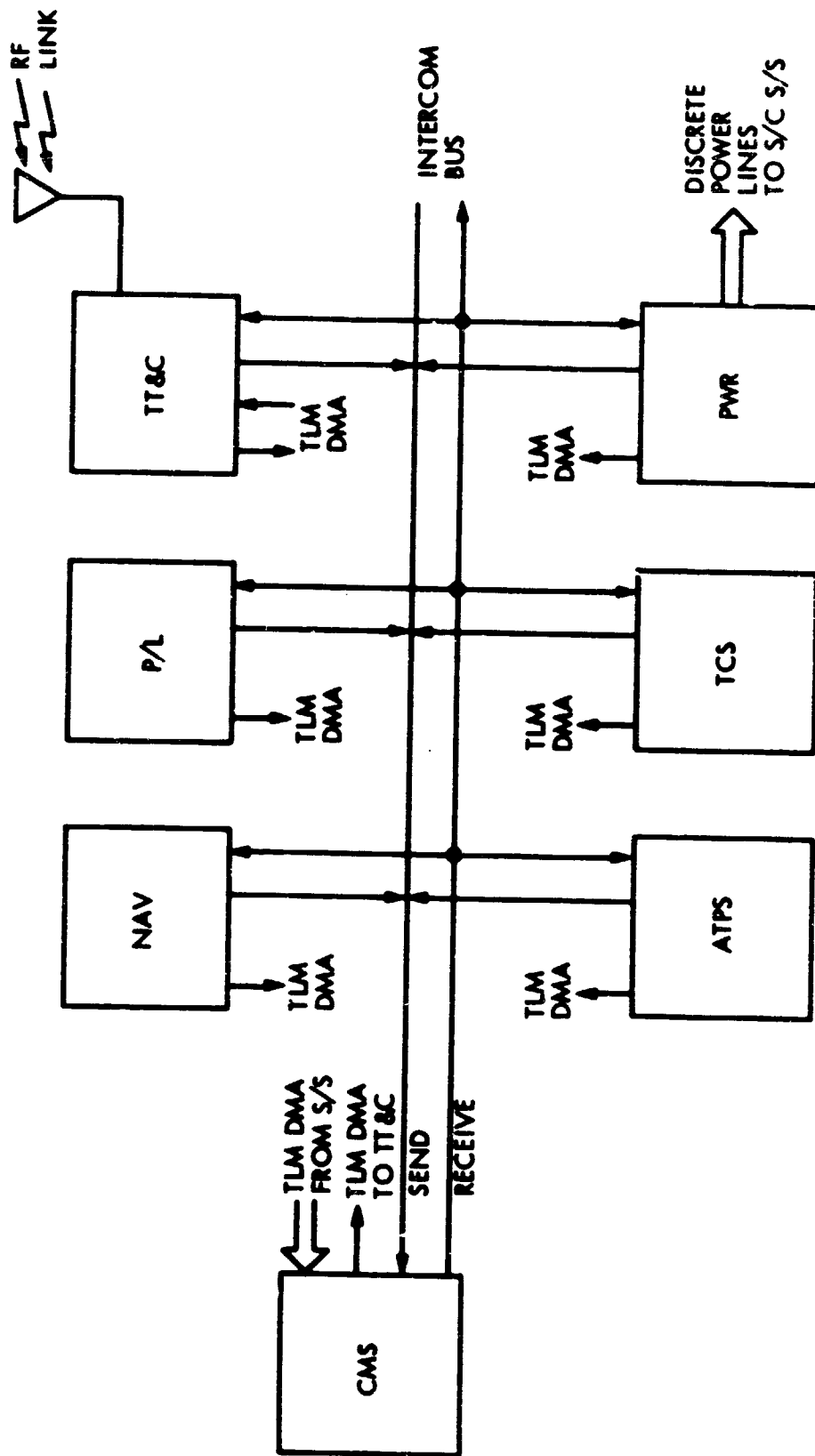


Figure D2-1. System Design Architecture Block Diagram

other subsystems, or decoded ground-issued commands transferred to the CMS from the TT&C. The service provided by the CMS to achieve information transfer between subsystems via the Intercom bus may be clarified through analogies. For example, CMS support of information transfer, which is specifically between only two subsystems, could be considered analogous to the service provided by a post office for its customers. A customer mails a letter, addressed to the post office box of another customer, at a single interface (the post office). The post office personnel then place the letter in the post office box of the addressed customer (analogous to filing received information at a specific address in the CMS memory). The addressed customer, by periodically interrogating his post office box, accesses the letter. The contents of the letter can be varied. For instance, it could provide information needed by the customer to repair his car (analogous to parametric data received from other subsystems). It may contain a request from a charity for a donation (analogous to a request from another subsystem). On the other hand, it might be a demand from the government to pay his income tax (analogous to a ground issued command).

For applications where the same information may be accessed from the CMS by several subsystems, a single CMS memory address, accessible by all interested subsystems, would be used to maximize CMS functional efficiency. This service would be analogous to posting a notice on a bulletin board. Received information from a specific source, when posted on a bulletin board, is available to all interested parties. Several interested parties interrogating the same bulletin board to access the posted information would be analogous to several subsystems interrogating the same memory address.

As part of the system design, time slot allocations are assigned to each subsystem for communications with the CMS via the Intercom bus. Such transfer of information from and to any particular subsystem must be accomplished within an allocated time slot for that subsystem. To minimize Intercom bus traffic needs and maximize information transfer efficiency, telemetry data transfer, which is a continuous demand function, is accomplished independent of the Intercom bus. In the distributed architecture of Figure D2-1, each subsystem acquires, multiplexes, and digitizes its own telemetry information. Referring to Figure D2-1, each subsystem uses a dedicated line to continuously provide its updated digital telemetry data via Direct Memory Access (DMA) to specifically allocated addresses in the CMS. It is also noted that the TT&C accesses, via DMA, the digital telemetry words from these CMS addresses as required to form the output telemetry stream. Since telemetry is a continuous function, this frees the Intercom bus so that it may be solely dedicated to interactive transfer of non-telemetric information, thereby significantly maximizing information transfer efficiency between subsystems.

It is also noted from Figure D2-1 that power is transferred to each subsystem via dedicated lines from the Power Subsystem (PWR). This would be in the form of a single dc voltage level. Therefore, each subsystem must internally generate the specific voltage levels and regulation required for its functional operation.

B. EVALUATION

1. Benefits

Several benefits may be realized from use of the decentralized architecture described in Section II.A.5 when compared with processing architectures requiring a centralized control function. Some of the more pertinent benefits are identified as follows:

a) Reduced Number and Complexity of Subsystem Interfaces

As more processing is relegated to the subsystem level, less information must be transferred to and from individual subsystems. Therefore, subsystem external interfaces become more simplified. For instance, a typical telemetry subsystem in a centralized architecture would normally require hundreds of analog signal interface lines from subsystems throughout the entire spacecraft. By decentralizing the processing function, the telemetry operations of analog signal acquisition, multiplexing, and analog-to-digital conversion are accomplished internally by each subsystem for its designated area of measurement responsibility. Therefore, the digitized telemetry measurement information from each subsystem can be provided as a serial stream of bits via a single digital interface line. Referring to the architecture of Figure D2-1, each subsystem, with the exception of the CMS, has a signal interface with only one other subsystem - the CMS. Furthermore, all signal interfaces are digital.

b) Increased Speed and Efficiency of Interactive Information Transfer Between Subsystems

As discussed in Section II.A.5, telemetry information transfer is handled by means of CMS DMA lines which are independent of the Intercom bus. The Intercom bus is therefore used to transfer only subsystem interactive information (data and commands) between subsystems. Since all non-interactive subsystem processing needs are accomplished internally in each subsystem, the volume of required information to be transferred between subsystems via the Intercom bus, even under crisis conditions, should be grossly reduced when compared to intersubsystem communications required in a centralized processing architecture. Therefore, the speed and efficiency of interactive information transfer between subsystems should be significantly better for the decentralized system architecture.

c) Increased Throughput Rate and Operational Efficiency

Throughput rate and operational efficiency increases in proportion to the number of processing functions that can be performed simultaneously using parallel processors. In a centralized architecture, the processing functions required by each subsystem must be time shared in a single computer. This severely limits the throughput rate and operational efficiency of the system since there is a practical limit in processing capability that is feasible from a single computer based on mass and power considerations. The decentralized architecture of Figure D2-1 dedicates a separate computer to each subsystem. Therefore, the processing requirements for all subsystems may be performed simultaneously. This significantly increases the possible throughput rate and operational efficiency of the system when compared with a centralized architecture. Furthermore, this overall improvement can still be realized using computer designs that are significantly less complex and demanding in power than that required for a centralized computer implementation.

d) Reduced System Integration Costs

Decentralized processing inherently allows considerable system independence for the test, validation, and operation of subsystems. If the integrity of the system interface requirements for a subsystem is maintained, that subsystem may be almost entirely tested and validated prior to system integration. In contrast, for centralized architectures, because of the comparatively more complex interface requirements and subsystem dependence upon the central processor, very little subsystem test and validation can be accomplished until each subsystem is integrated into the complete system. Therefore, the normally large costs attributed to the spacecraft system integration, test, and operation phases for missions employing centralized system design architectures should be significantly reduced by the use of the decentralized system design architecture of Figure D2-1.

e) Increased Multimission Applicability

An inherent feature of the decentralized processing architecture described in Figure D2-1 is multimission applicability. In contrast, a centralized system design architecture tends to be mission dependent since the performance capability of the central computer has profound effects on the operating limitations of all subsystems. Decentralized processing, in general, allows considerable system independence for the internal design and operation of subsystems assuming the integrity of subsystem external interface requirements are maintained. The subsystem signal interface requirements for the architecture of Figure D2-1 involve simply writing into and reading from CMS memory. Therefore, entire subsystem internal designs could be changed and readily accommodated by the system. Furthermore, old subsystems could be deleted from and new subsystems added to the Intercom bus of Figure D2-1 with no significant

perturbations to the overall system design. In like manner, new mission requirements and priorities could be readily accommodated through reallocation of CMS memory space.

f) Increased Growth Potential

The system design architecture of Figure D2-1 inherently provides high growth potential. The internal processing capability of individual subsystems can be significantly increased with little effect on the system design architecture. This is primarily due to the fact that each subsystem can simultaneously perform its processing functions in parallel with other subsystems. Furthermore, more subsystems can be added to the Intercom bus, limited only by bus traffic capability and CMS memory capacity. Since 1) the Intercom bus traffic only involves interactive information transfer between subsystems and 2) internal subsystem processing greatly reduces the need for such external information transfer, high system throughput rates and operational efficiency characteristics can be realized. In contrast, the growth potential is considerably more limited for a centralized architecture since the processing and control requirements for all spacecraft subsystems must be accomplished by a single computer on a time-shared basis placing practical limitations on achievable throughput rates and operational efficiencies.

2. Performance Features

Two key D2-1 performance features of the decentralized architecture of Figure 1 that result from the benefits discussed in Section II.B.1 are defined as follows:

a) Increased On-Board Processing Capability

Compared with a single central computer doing all of the processing, the spacecraft processing is distributed to many less complex dedicated computers working in parallel. This circumvents the limited throughput rate problem associated with a single shared computer. Since processing for every spacecraft subsystem can be performed simultaneously rather than on a serial time-shared basis, higher system operational efficiency can also be achieved.

b) Reduced Response Time for Critical System-Level Decisions

Since more processing is allocated to the subsystem level in the decentralized architecture, the amount of interactive information that must be transferred between subsystems is greatly reduced over that required for a centralized system. Therefore, a proportional increase in information transfer rates between subsystems may be realized. Furthermore, the increased processing capability provided by the decentralized architecture, discussed in Section II.B.2(a), also contributes to faster information transfer rates. The faster information transfer rates provided by the decentralized system design architecture result in better response times for accomplishing critical on-board decisions.

3. Flexibility

The decentralized system design architecture of Figure 1 provides an extremely high level of flexibility when compared with alternative architectures using centralized processing and/or control. Increased flexibility is realized in the following areas:

a) Types and Number of Subsystems

The system design architecture of Figure D2-1 can readily adapt to changes in both the types and the number of subsystems. Since the only signal interface for each subsystem is with the CMS and this interface involves only a memory write and/or read function, subsystem changes could normally be accommodated by reallocation of CMS memory addresses. Significant increases in spacecraft subsystem sophistication in both type and number could be accommodated by modular increases in CMS capacity and Power Subsystem (PWR) capability.

b) Subsystem Redesign

Assuming functional system interface requirements are maintained with the CMS, entire subsystem designs can be internally changed with minimal effect on the system design. This provides considerable freedom for the subsystem designer to modify the internal design of a subsystem if necessary with very little impact upon the other subsystems or the integrated system.

c) Mission Independence

The decentralized system architecture of Figure D2-1 is adaptable to a wide range of mission requirements. Since individual subsystem designs are flexible, it is assumed that they will be internally designed to efficiently accommodate the processing needs of the mission for which they are used. Furthermore, all processing is accomplished at the subsystem level. Therefore, subsystems for a variety of missions can be integrated into the processing design architecture of Figure D2-1 with virtual independence of mission application at the system level. This is not true for a centralized system, since the architectural design of the central processor, and therefore its system-level interfaces with the individual subsystems, are driven by mission-unique requirements.

4. Constraints

In evolving the decentralized system design architecture of Figure D2-1, several constraints, based upon an assessment of architecture-related needs for achieving a viable system design, have been defined as follows:

a) Subsystem Nonvolatile Memory

Each subsystem must provide an appropriate level of nonvolatile memory capability to protect against loss of critical internal software due to temporary interruptions in power.

b) Central Timing

The system design architecture must provide a common timing signal to all subsystems. This signal will have to be distributed by the CMS since it is the only subsystem that has signal interfaces with all subsystems.

c) Time Slot Allocations

A subsystem can communicate with the CMS via the Intercom bus only during specific time slots allocated to that subsystem as part of the system design.

d) Early Top-Down System Design

Since there is no central executive control function, an early top-down system design effort is required for each mission so that the executive control software functions may be properly allocated between the subsystems prior to detailed subsystem design activities.

5. Key Tradeoff Parameters

To accomplish the best overall system design for the decentralized architecture of Figure D2-1, commensurate with the needs of a particular mission, the following tradeoffs should be performed.

a) Level of Shared Versus Dedicated Sensor Allocation Between Subsystems

Some subsystems require the same parametric information to perform some of their allocated functions. For example, NAV may have sensors from which it derives orbital information needed by ATPS. ATPS may either acquire this information by 1) interrogating the information provided by NAV from specific addresses of the CMS or 2) providing its own sensors to derive the desired information itself. Tradeoffs between information transfer time response requirements, system complexity, subsystem complexity, mass, power, reliability and cost should be evaluated to achieve the best level of balance commensurate with mission needs.

b) Functional Partitioning of Responsibility Between Subsystems Versus System Performance Needs

Since the decentralized system design architecture has no central processing and/or control function, all responsibilities for processing and control must be distributed between subsystems. This requires a fresh look at who should do what when compared with a centralized architecture. For example, a subsystem that physically positions an antenna in a centralized architecture may not be the best subsystem to perform that function in a decentralized architecture. Certainly, anything accomplished by a centralized architecture, with or without distributed processing, can be accomplished by a decentralized architecture. Both architectures contain CPU and memory. What is done by the CPU and memory is really controlled by the software programmer on the ground. The

question is not whether a decentralized architecture can do the job, but how efficiently it does the job. To achieve maximum efficiency, it is necessary to properly partition functional responsibility between the subsystems to best meet the system performance requirements. This will require tradeoffs between sensor locations, information transfer time response requirements, system complexity, subsystem complexity, mass, power, reliability and cost.

c) Level of Fault Tolerance Versus Implementation Complexity and Cost

The requirement for each subsystem of a decentralized system design architecture to be fault tolerant implies a self-checking fault tolerant computer in each subsystem. This is necessary, since there is no central computer to perform fault detection diagnostics on and effect fault correction of the subsystems. Since the subsystem computer must be both self-checking and self-correcting, implementation complexity and cost limit the practical levels of fault tolerance achievable with a given technology for any particular mission. Tradeoffs between the level of fault tolerance, subsystem complexity, and cost should be performed to determine the optimum balance commensurate with available technology, acceptable risk, and overall mission needs.

6. Other Drivers

Several system design parameters that could be significant drivers in affecting both the performance and implementation characteristics of the decentralized system design architecture of Figure D2-1 are identified as follows:

a) Data Transfer Efficiency Between Subsystems

This is an important characteristic for any system design architecture regardless of how it is effected. The decentralized system design architecture of Figure D2-1 has offloaded the telemetry function from the subsystem Intercom bus to limit the traffic to subsystem interactive needs only. Furthermore, the subsystem interactive information transfer requirements are significantly reduced from that for a centralized architecture due to the increased level of processing accomplished within each subsystem. Since Intercom bus traffic requirements should be very low, it has been assumed that fixed time slot allocations for subsystem communications over the Intercom bus shall suffice for the highest information transfer rate requirement of any subsystem under all mission conditions. On the other hand, it may not. Under such circumstances, adaptive time slot allocation may be necessary.

b) Efficiency of CMS Memory Allocation

The methodology used for allocation of memory space within the CMS to meet the system and subsystem performance needs is a potentially critical driver. It could have a significant effect on the implementation characteristics of the CMS with respect to mass and power. Furthermore, it can drive both system and subsystem

performance characteristics. For instance, inefficient memory allocation procedures could significantly decrease the effective information transfer rate between subsystems. As noted in Section II.A.5, procedures, such as storing information to be accessed by several subsystems in a single CMS memory address, should be employed to maximize CMS functional efficiency.

- c) Complexity of Subsystem Fault Routine: Associated with System Related Fault Diagnostics and Correction

For the decentralized system design architecture of Figure D2-1, the system-level functions of fault detection and correction where more than one subsystem is involved must be performed by a set of software fault routines which are distributed among memories of the various spacecraft subsystems. The complexity required for the distributed fault routines, to interactively work together, to perform the system-level fault detection and correction functions should be defined, understood, and evaluated. The level of subsystem software routine complexity to perform the system-level functions required of the decentralized system design architecture of Figure D2-1 versus complexity of the software routines to accomplish the comparable functions in a centralized architecture is yet to be determined.

7. Detractors

Three items have been identified that represent potentially negative attributes of the decentralized system design architecture described in Section II.A.5. The potential significance of these items when compared with the characteristics of a centralized architecture is yet to be determined. Furthermore, mission complexity and technology development will heavily influence such a determination. The aforementioned items that could detract from the positive aspects of the subject architecture are defined as follows:

- a) Increased Criticality of an Early Top-Down System Design

Subsystem designs will, by the very nature of the decentralized system design architecture, be relatively independent of the system. However, complete independence is not practical since some system-level decisions must be made where interaction between subsystems is involved and/or required. This can be accomplished without a centralized computer by distributing the system-level executive software responsibilities to the appropriate subsystem computers. However, it requires an early system design effort in which the subsystem responsibilities for meeting the system-level needs are properly allocated and well defined. If this is not adequately done, it could potentially impact several subsystem computer software designs later in the program as opposed to one central computer software design in the centralized architecture. On the other hand, if properly done early in the program, the overall system design effort should be comparable to that required for a centralized system and a high degree of subsystem design independence should be practical.

b) Increased Number of Self-Checking Fault Tolerant Computers

For a centralized system design architecture, only one computer needs to be self-checking with regard to fault tolerance. For the decentralized system design architecture, each subsystem has its own computer with no external spacecraft executive control. Therefore, each such subsystem computer must be capable of checking itself and effecting its own corrective action. Certainly, the complexity of a subsystem computer would be significantly less than the complexity of a centralized computer capable of performing the total job that is distributed among several subsystem computers. In that case there may not be a disadvantage when compared with a fully centralized system design architecture. However, one could consider a distributed architecture in which a relatively small central executive performs the diagnostics and fault correction for each of the distributed subsystem computers via a common bus. In that case, only the central executive computer must take care of itself. Therefore, the subsystem computer designs realize a mass and power advantage over that required for the fully decentralized system design architecture. This advantage could be erased, for all practical purposes, by continued advances in technology. For instance, application of VLSI technology could allow cost effective implementations characterized by low mass and power.

c) Increased Complexity of Subsystem Designs

When part or all of the processing functions allocated to a centralized computer are distributed to the subsystems, the recipient subsystems have more functions to perform and therefore must become more complex. The increased subsystem complexity can potentially result in increased subsystem test time, lower subsystem reliability, and higher subsystem costs than required for a centralized system design architecture. The impact of this complexity increase is a function of available technology as noted in Section II.B.7.

8. Conclusions

The foregoing evaluation of the candidate decentralized system design architecture described in Section II.A.5 has resulted in the following preliminary conclusions:

a) Technical Feasibility

A fully decentralized (no central spacecraft executive control) system design architecture for spacecraft computer processing, capable of an overall performance level comparable to or greater than that possible from centralized architectures, appears technically feasible.

b) Early System Design

A concentrated and thorough system design effort will be required early in any program that uses a decentralized architecture. An efficient subsystem intercommunications media such as the CMS must be provided. System-level software responsibility must be properly distributed between the subsystems to insure acceptable performance of system-level functions through a cooperative subsystem effort.

c) Subsystem Test and Validation

A high degree of subsystem test and validation may be accomplished prior to system integration providing the potential for significant mission cost savings compared with centralized architectures.

d) Multimission Applicability

The fully decentralized processing architecture is more adaptable to multimission applications than centralized architectures of comparable performance level.

e) Further Evaluation

A more detailed tradeoff study to properly evaluate the potential of a fully decentralized processing architecture appears warranted.

III. CMS DESIGN ARCHITECTURE

A. DESCRIPTION

1. Function

The primary function of the Common Memory Subsystem (CMS) of the decentralized system design architecture is to provide an intermediate storage media for information transfer between other spacecraft subsystems. A secondary function of the CMS is to distribute a common timing signal to all spacecraft subsystems.

2. Functional Elements

The CMS consists of the following four functional elements:

- a) Self-Checking Fault Tolerant Computer
- b) Nonvolatile Buffer Memory
- c) Nonvolatile Mass Memory
- d) Input/Output Unit

3. Functional Requirements

The functional requirements imposed upon the CMS by the decentralized system design architecture are defined as follows:

- a) Receive and file updated spacecraft information (parametric data, requests, and commands) when provided by other subsystems.
- b) Provide specific stored spacecraft information (parametric data, requests, and commands) to other subsystems when requested.
- c) Store spacecraft audit trail data and critical spacecraft software routines.
- d) Provide self-maintained fault tolerance to all internal single-point failures.

4. Interface Requirements

The interface requirements imposed upon the CMS by the decentralized system design architecture are defined as follows:

- a) Provide direct memory access (DMA), via dedicated lines from each subsystem, for the purpose of receiving subsystem telemetry data.

- b) Provide a dedicated DMA line to TT&C for reading out digital telemetry words when requested by TT&C.
- c) Provide a digital Intercom bus interface with each subsystem for writing data in and reading data from CMS memory.
- d) Accommodate a dedicated power input line interface with PWR.

5. Block Diagram

A functional block diagram for a candidate CMS design architecture is given in Figure D2-2. Referring to Figure D2-2, all data to and from all other spacecraft subsystems is routed through the Input/Output (I/O) unit. This includes DMA for telemetry data transfer and Intercom bus traffic for the transfer of interactive information between subsystems.

All spacecraft data transfer, with the exception of audit trail readout to the ground, is accomplished through a volatile read/write memory in the Self-Checking Fault Tolerant Computer block of Figure D2-2. This is the main working memory having high-speed random access capability. As noted in Figure D2-2, there are two additional levels of memory - the nonvolatile buffer memory and the nonvolatile mass memory. The nonvolatile buffer memory interfaces directly with the computer memory providing a slower access speed but greater capacity than the volatile computer block memory. It stores critical software routines and buffers blocks of audit trail information. The nonvolatile mass memory receives and stores the blocks of audit trail data from the nonvolatile buffer memory. It provides long-term storage of audit trail data for extended periods of autonomous operation. As noted from Figure D2-2, the nonvolatile mass memory may be accessed directly by the ground through the I/O unit. Also, any data or software stored in either the nonvolatile buffer memory or the nonvolatile mass memory may be accessed by the computer memory for transfer to any spacecraft subsystem as required.

Referring to Figure D2-2, the Self-Checking Fault Tolerant Computer block provides the fault detection, fault isolation, and correction command issuance for not only itself but the remaining blocks of the CMS. This includes the nonvolatile buffer memory, the nonvolatile mass memory, the I/O unit, and the power converter unit, all of which are block redundant.

Typical implementation characteristics might reflect an 8 kiloword to 32 kiloword Complementary Metal Oxide Semiconductor (CMOS) computer memory, a 10^6 bit to 10^7 bit bubble memory for the nonvolatile buffer, and a 10^8 bit to 10^9 bit tape recorder for the nonvolatile mass memory.

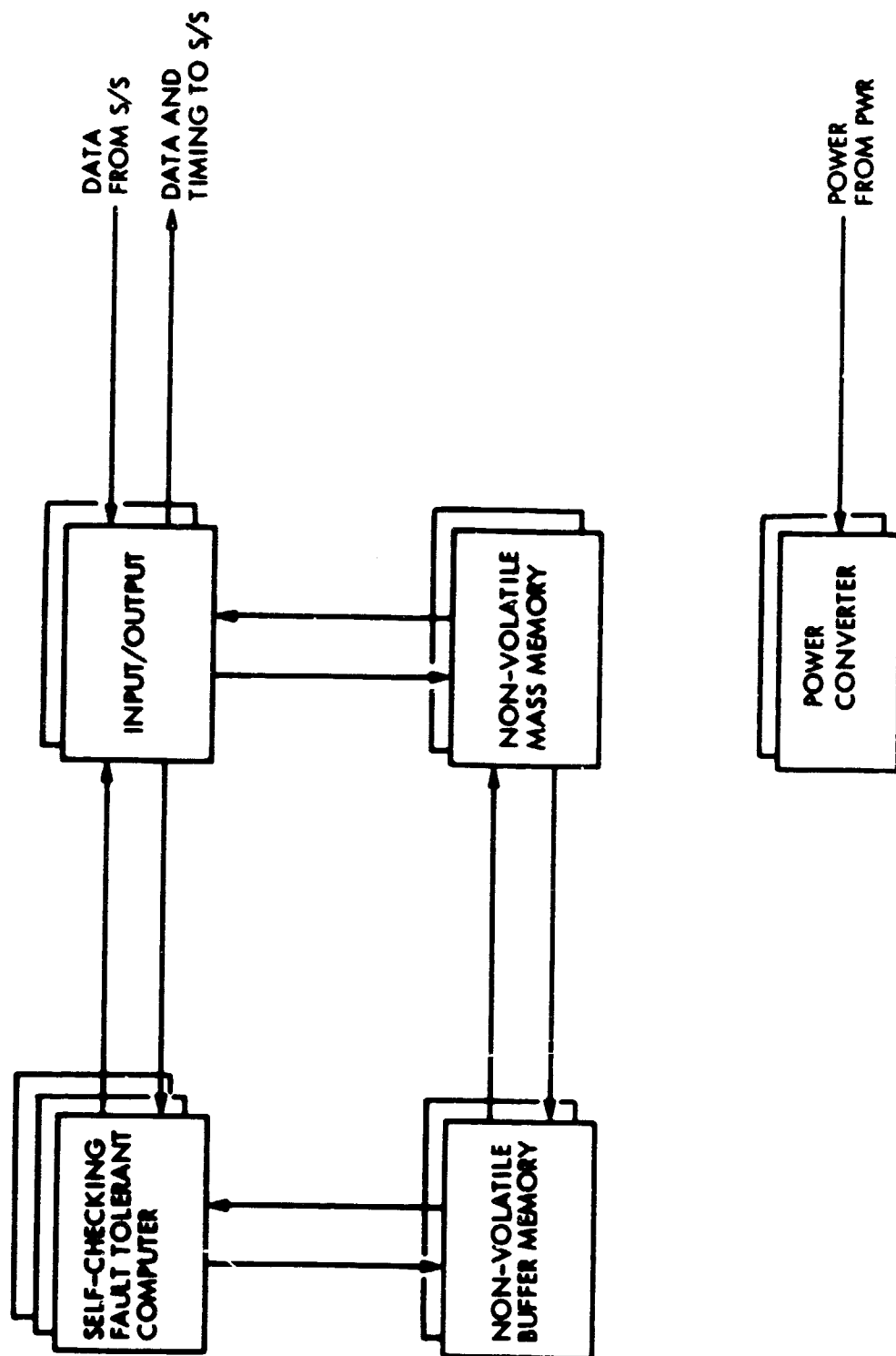


Figure D2-2. CIS Functional Block Diagram

B. EVALUATION

1. Benefits

Potential benefits that could be realized from use of the CMS in a decentralized system design architecture are identified as follows:

a) Utilizes Existing Subsystem Capability

A buffered mass data storage subsystem comparable to the CMS design described in Section III.A.5 is required to accommodate the audit trail storage for autonomous spacecraft missions independent of the system design architecture. In the decentralized architecture this existing data storage capability could be modified to become the CMS and used to accomplish transfer of information between subsystems. Therefore, the media for the transfer of information between subsystems is virtually free for the decentralized architecture.

b) Eliminates Need for Central Executive Subsystem

The CMS eliminates the need for a separate and relatively complex subsystem to perform the central executive control function between subsystems.

2. Performance Features

a) Capable of High Speed Random Access

Through use of a comparatively low capacity, volatile, random access read/write memory for direct subsystem interfacing, high speed information transfer between subsystems can be achieved as required.

b) Capable of Protecting Critical Information

Nonvolatility is provided in a comparatively high capacity internal memory to protect 1) audit trail data being accumulated for both spacecraft and ground interrogation and 2) critical software routines to be used for reloading spacecraft memories as required.

c) Passive Operation

Information transfer between subsystems can be achieved passively (no generation and issuance of external commands) by simply writing into and reading from specific memory addresses as requested by the subsystems.

3. Flexibility

Characteristics of the CMS related to its flexibility are described as follows:

a) Performance Adaptivity

Through application of multiple technologies (CMOS, bubble, and tape), the elements of the CMS can be readily organized to provide both high information transfer rates and high storage capacity at a comparatively low average power level.

b) Capacity Adaptivity

The storage capacity of all memories can be easily increased or decreased to best meet the needs of any particular mission by use of a modular approach to memory design and implementation.

c) Mission Independence

The specific information transfer needs of any mission and/or mission phase may be readily accommodated through reallocation of memory space and subsystem time slot availability.

4. Constraints

Pertinent constraints levied on the CMS when used in a decentralized system design architecture are identified as follows:

a) Fixed Time Slot Allocation

The CMS can access information from or provide information to a specific subsystem only during predefined time slot allocations for that subsystem.

b) Self-Imposed Integrity Maintenance

Since there is no central spacecraft computer to perform fault diagnostics on and effect fault correction of the CMS, the CMS computer must be both self-checking and self-correcting. It must also perform the fault detection and recovery functions for the remaining block redundant elements of the CMS. These consist of the nonvolatile buffer memory, the nonvolatile mass memory, and the I/O unit.

5. Tradeoffs

The following tradeoffs, related to use of the CMS in a decentralized system design architecture should be performed prior to a CMS detailed design.

a) Adaptive Time Slot Allocation Versus Complexity and Cost

Tradeoffs between the feasibility of providing adaptive time slot allocation to subsystems as a function of mission need versus complexity and cost should be performed.

b) Level of Fault Tolerance Versus Implementation Complexity and Cost

Tradeoffs between the level of fault tolerance, subsystem complexity, and cost should be performed to determine the optimum balance commensurate with available technology, acceptable risk, and overall mission needs.

6. Other Drivers

Some CMS design-related drivers that could significantly affect the performance and implementation characteristics of the CMS when used in a decentralized system design architecture are defined as follows:

a) Memory Allocation Methodology

The methodology used for memory space allocation within the CMS is a potentially critical driver. It could have a significant effect on CMS performance, reliability, mass and power.

b) Available Technology

The technology available for implementation of the various memory blocks of the CMS will have a profound effect on the performance, reliability, mass, and power characteristics of the CMS.

7. Detractors

Some potentially negative attributes of the CMS design architecture defined herein are discussed as follows:

a) Dependence on Several Technologies

The CMS design architecture defined in Section III.A.5 would require three different types of memory technology for its implementation due to the wide range of performance capability required. These are CMOS, bubble, and tape. Furthermore, serial implementation of the different technologies is also required thereby potentially decreasing reliability.

b) Dependence on Electromechanical Devices

Assuming current state-of-the-art technology, the anticipated mass storage requirements for the CMS design architecture defined in Section III.A.5 dictate the use of a magnetic tape recorder to achieve the necessary capacity. Since tape recorders have mechanical wear-out characteristics, stop/start cycles must be limited by the use of a large capacity buffer. If the technology used for the solid-state nonvolatile buffer could be extended to encompass the storage capacity required for the nonvolatile mass memory, the tape recorder could be eliminated.

8. Conclusions

The foregoing evaluation of the candidate CMS design architecture described in Section III.A.5 has resulted in the following preliminary conclusions:

a) Information Transfer Effectivity

The candidate CMS design architecture appears to provide an efficient means of information transfer between fully decentralized subsystems where no central spacecraft computer control is employed.

b) The candidate CMS design architecture may be used to effect executive control over one or more subsystems if desired with nothing more than software modifications.

c) Technology Requirements

The candidate CMS design architecture may be implemented using only current state-of-the-art technology so that no advanced technology development is required.

IV. TT&C DESIGN ARCHITECTURE

A. DESCRIPTIONS

The TT&C receives, processes and transmits information (data) to and from the spacecraft over RF links. The TT&C receives, tracks, demodulates, detects, conditions and processes data received over the RF "uplink". The TT&C also accepts data from the spacecraft CMS, processes the data, conditions it, modulates it on an RF carrier, amplifies the modulated RF carrier and transmits the data modulated carrier on the RF "downlink". Some data is received on the uplink and is appropriately conditioned and transmitted on the downlink for two-way data transfer. These processes are performed by the three major RF functional elements - the Uplink, the Two-Way and the Downlink.

1. Function

The TT&C provides telecommunication functions for the spacecraft. Over the uplink, the TT&C subsystem receives data modulated on an RF carrier. The command data is demodulated from the carrier and the data bits detected. This data is then processed and sent to the Common Memory Subsystem (CMS) for storage and subsequent use by other subsystems. Some received data is two-way or turnaround data which is processed in the TT&C RF equipment and then transmitted on the downlink. Spacecraft data accumulated in the CMS memory is processed, formatted, and transmitted on the downlink by the TT&C. The TT&C functional block diagram is shown in Figure D2-3. This block diagram shows the three major functional elements: the Uplink (Up), the Two-Way (Tw) and the Downlink (Dn). The Antennas, Antenna Control, Antenna Select, Microwave Components, and the Control and Monitor are functional elements of the TT&C used to accomplish the three key functions.

2. Functional Elements

The following represents a partitioning of the TT&C into functional elements:

a) Uplink Function

The functional block diagram for the uplink function (Up) with its functional elements is shown in Figure D2-4. The uplink function receives information from data modulated on an RF carrier (e.g., from the ground or another satellite). One or more of the antennas receive the RF carrier. The antennas may require steering (mechanically or electrically) to point to the signal source. The steering may be from signals generated within the TT&C or by the Attitude, Translation and Pointing Subsystem (ATPS). The received RF signal is filtered and directed by the antenna select and microwave component elements. The RF signal is then down converted in frequency; the data is demodulated; the data information is detected; the data is conditioned;

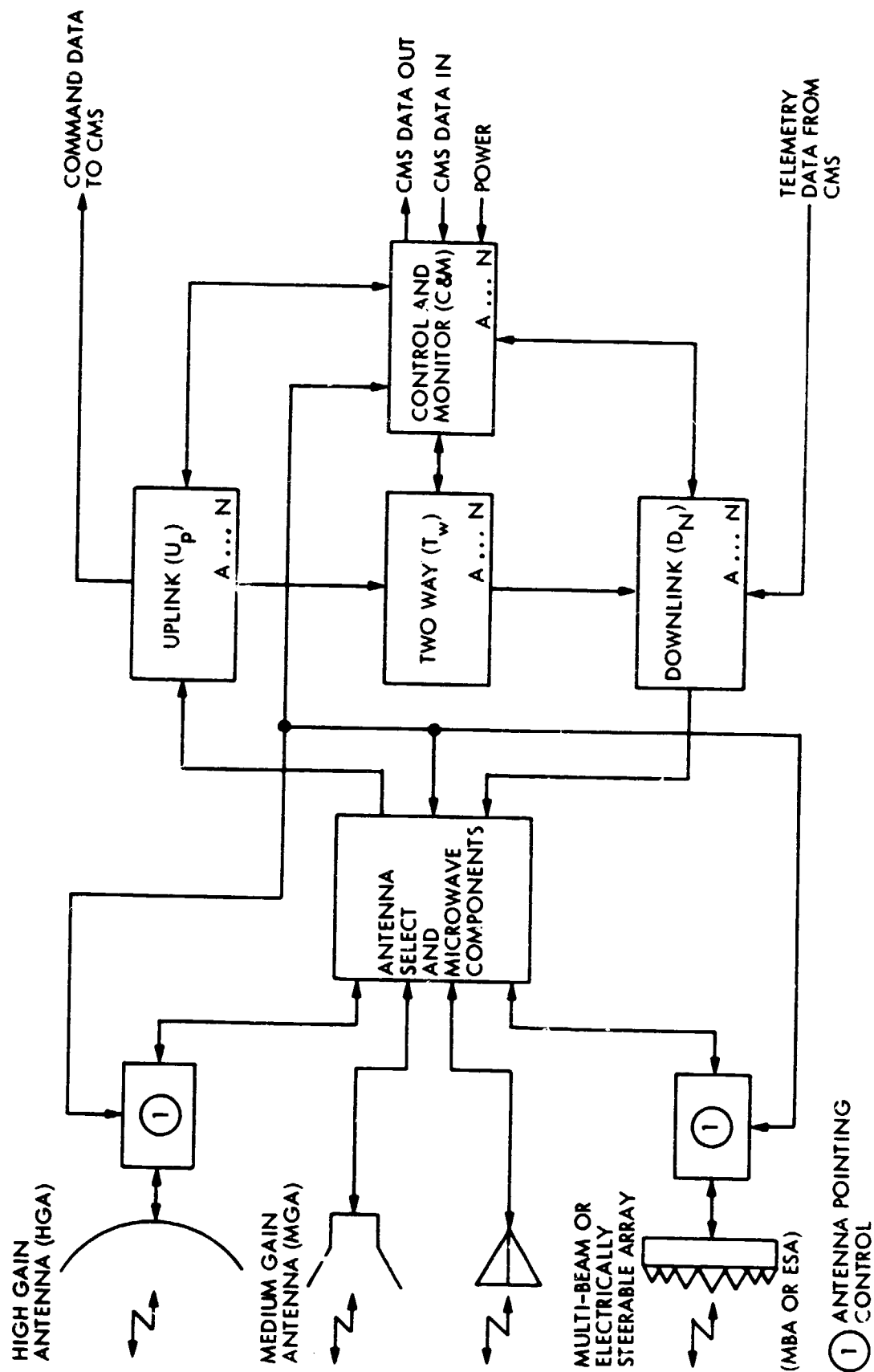


Figure D2-3. Generalized TT&C Functional Block Diagram

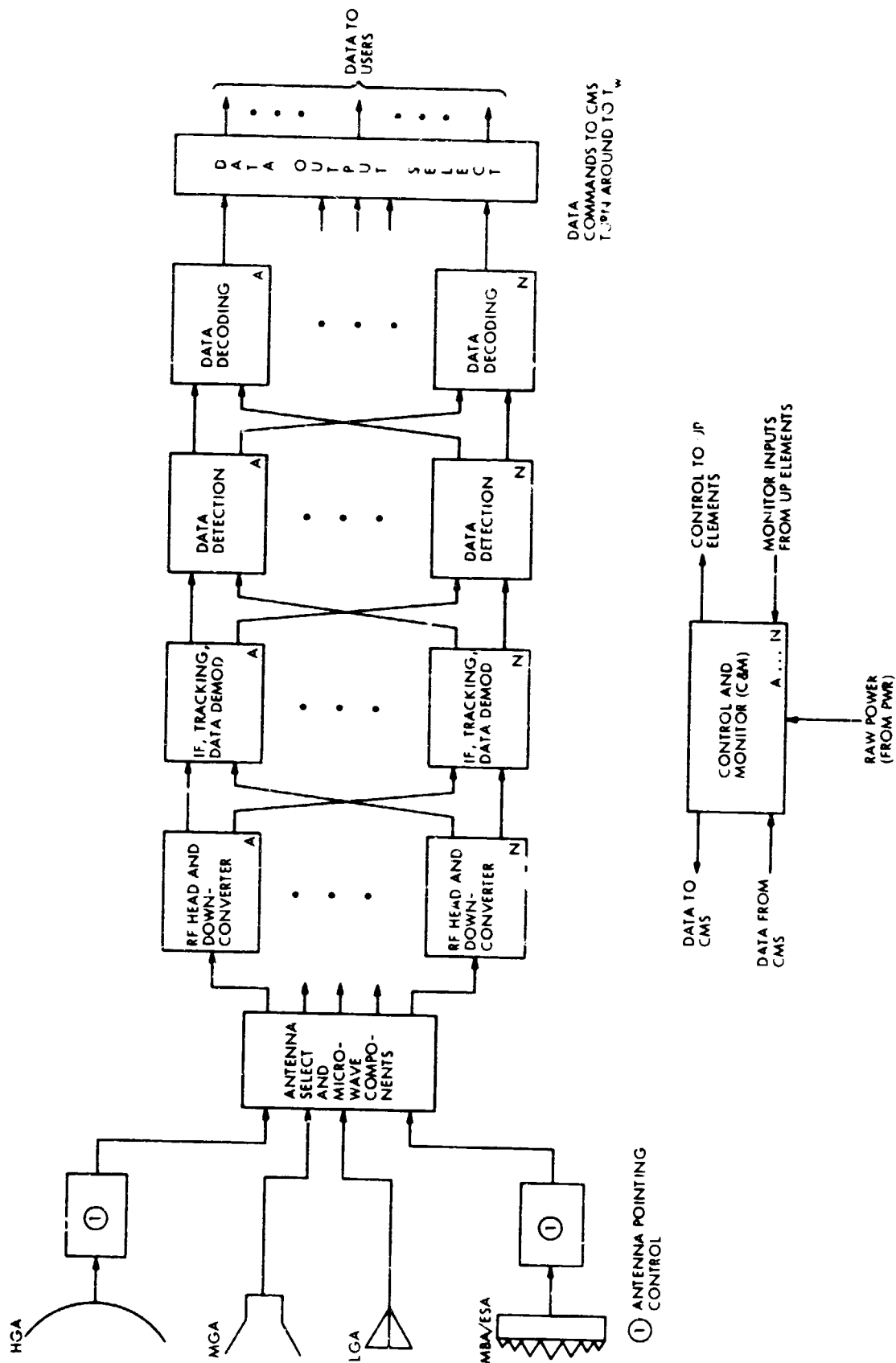


Figure D2-4. Generalized TT3C Uplink (U_p) Functional Block Diagram

and, the data is decoded as command information to be sent to the CMS. Some of the uplink data is conditioned and turned around for two-way transmission on the downlink.

The control and monitor functional element monitors the TT&C status and via the CMS receives other information required to maintain and provide the uplink function service.

b) Downlink Function

The downlink function (D_N) and its functional elements are shown in Figure D2-5. The downlink function obtains data from the CMS memory or two-way function at its input. The data is formatted and processed; then it is conditioned and modulated on the RF carrier. The modulated RF signal is amplified and provided to the antenna selector and microwave components for transmission over one or more of the antennas. Some of the antennas may be steerable and/or beam formable. These antennas receive pointing direction from the TT&C.

The control and monitor functional element accepts internal and external information required to maintain and provide the downlink function service.

c) Two-way Function

The two-way function (T_W) is shown with its functional elements in Figure D2-6. The two-way function accepts data/information from the uplink function and conditions it. This conditioned data is then provided to the downlink function for retransmission (e.g., the data could be the carrier and/or ranging for two-way tracking).

The control and monitor functional element accepts internal and external information required to maintain and provide the two-way function service.

3. Functional Requirements

The functional requirements imposed upon the TT&C by the decentralized system design architecture are defined as follows:

- a) Receive an RF carrier with data modulated on it. (This is over the uplink* path.)

*NOTE: The terms uplink and downlink are used figuratively. These are the normal terms used for the RF paths over which signals are received by and transmitted to and from the spacecraft. (These RF paths could be between any sources, e.g., the spacecraft and another satellite.)

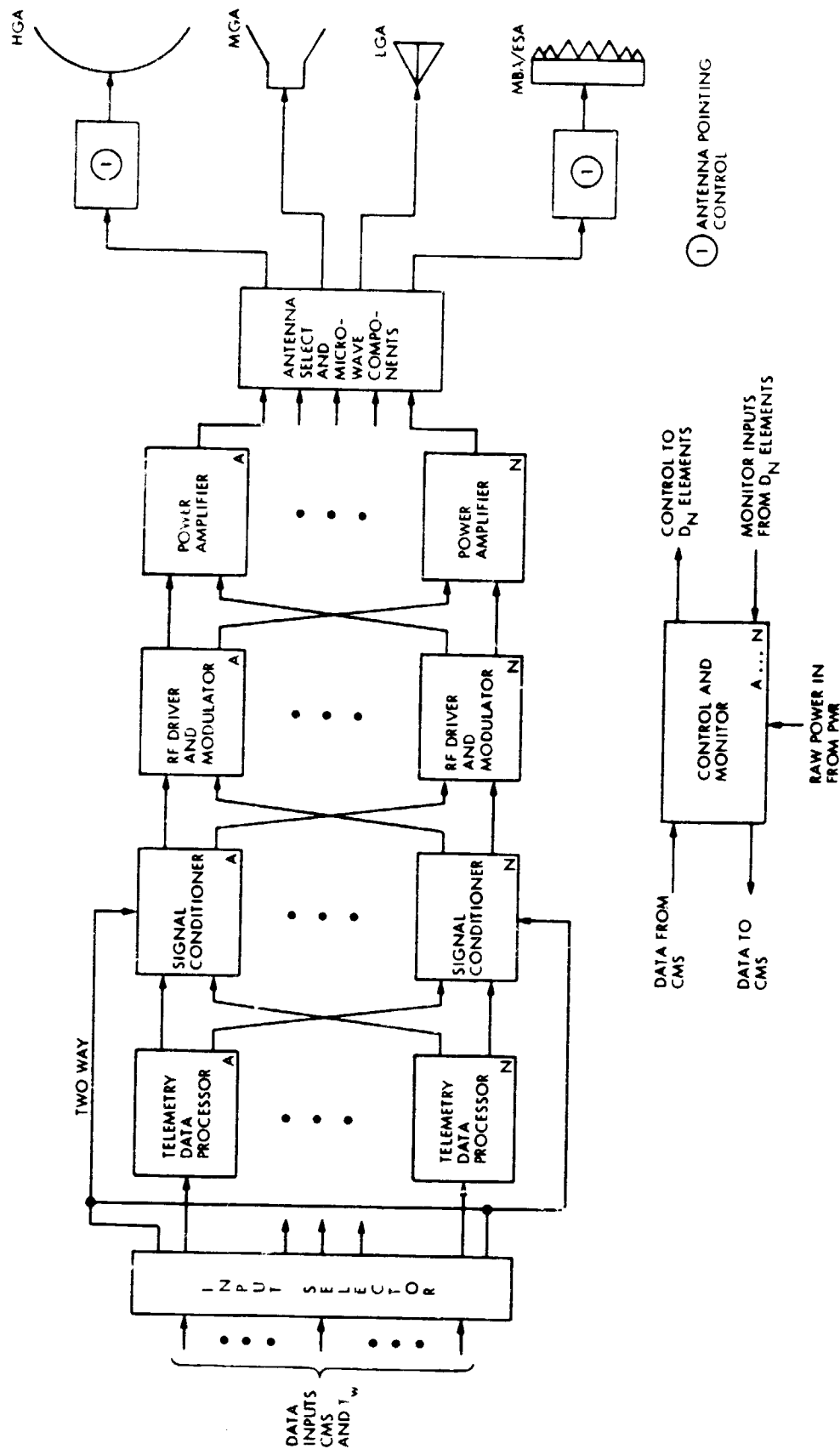


Figure D2-5. Generalized TT&C Downlink (D₁₁) Functional Block Diagram

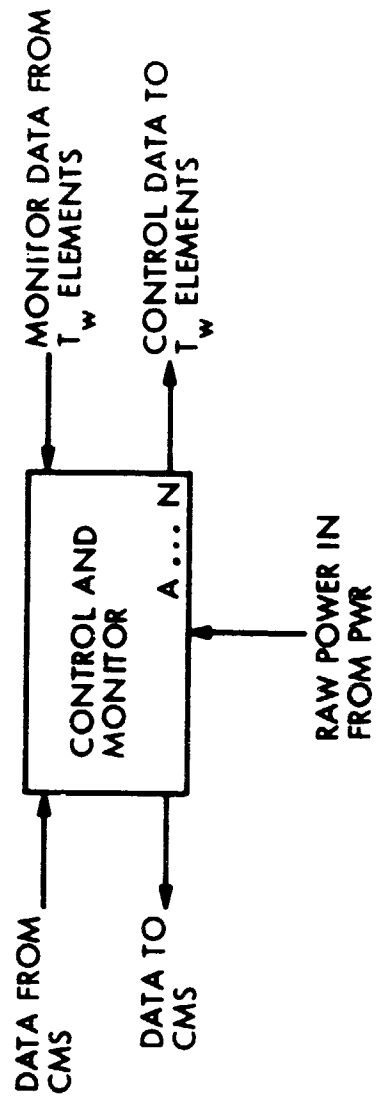
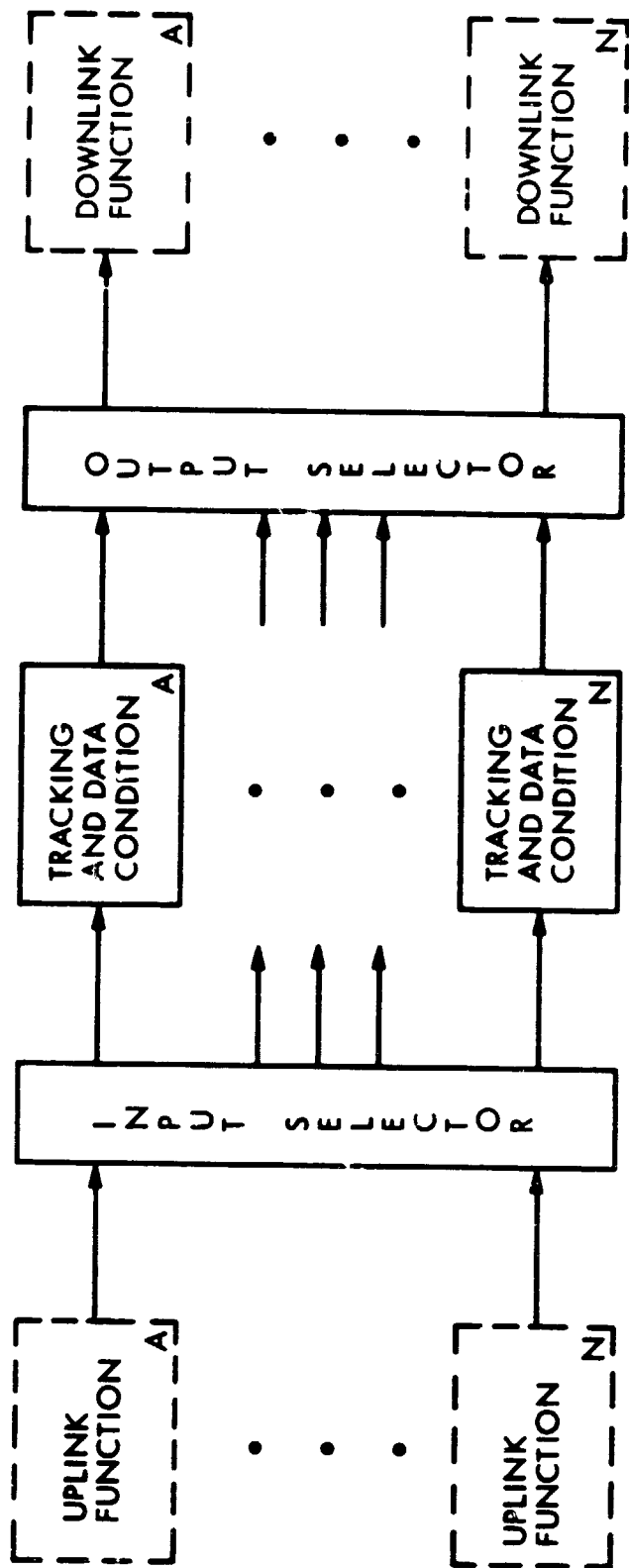


Figure D2-6. Generalized TT&C Two Way (T_w) Functional Block Diagram

- b) Demodulate and detect the information modulated on the carrier.
- c) Condition the uplink detected data bits into an information bit stream.
- d) Decode and validate the command information words.
- e) Provide decoded and validated command words to the CMS.
- f) Demodulate and detect two-way data on the uplink carrier; translate the data; modulate the data on the downlink carrier; and, transmit the modulated carrier.
- g) Receive data from the CMS memory. This data shall be formatted into a telemetry data stream. The formatted data shall be modulated on an RF carrier and transmitted via the downlink path from the spacecraft.
- h) Provide single-point failure fault tolerance. The uplink, downlink and two-way functions shall meet their specified requirements with one failure. Wherever practicable, the design shall provide graceful or no degradation with more than one failure. The ability to survive a single failure and the ability of the TT&C design to meet requirements with multiple failures are mission dependent functions. (However, to achieve level 5 autonomy, the meeting of the one failure criteria is probably a minimum). The key to the no single-point failure design is the control and monitor function. This function will have to incorporate a fault tolerant self-checking computer capability.
- i) Maintain "availability" of the uplink function for reception of data. In the event of a failure, this function shall be available within a "reasonable" amount of time. This function could be unavailable for "brief" periods of time while self health and maintenance checks are being performed. The terms "reasonable" and "brief" are mission defined durations.
- j) Maintain availability and/or operation of the downlink function. The downlink function shall be available to return data or shall be sending data almost continuously. This service can be temporarily interrupted during a failure or during health and maintenance checks. The duration of failure down time and health checks are mission derived parameters.
- k) Maintain availability of the two-way function for use on the same basis as the uplink and downlink functions.

4. Interface Requirements

The interface requirements imposed upon the TT&C by the decentralized system design architecture are defined as follows:

a) Power Subsystem (PWR)

- 1) The TT&C receives redundant power from PWR.
- 2) The power is raw. The TT&C converts this raw voltage for internal use.
- 3) Power management is performed by the TT&C on the incoming power. The TT&C optimizes the allocated power within preset but variable states (e.g., maximum power, mission phase, spacecraft states).
- 4) The TT&C controls the power based on information from internal and external sensors and information.
- 5) Typically the downlink function is one of the major power users on a spacecraft. This function typically will have separate power lines or sources.
- 6) Emergency standby power shall be provided by PWR to maintain critical mission functions (typically for critical memory or data storage and preserving the uplink command capability).

b) Common Memory Subsystem (CMS)

- 1) The TT&C can request and receive preprogrammed stored sequences from the CMS memory. However, some or all of these could be stored in the TT&C memories.
- 2) The TT&C will receive executive or direct commands from the mission control via the CMS (i.e., ground executive control).
- 3) The TT&C provides to the CMS all status, sensor data, and telemetry type data required by the spacecraft subsystems.
- 4) The TT&C receives, when required, any subsystem or spacecraft information, status or sensor data from the CMS.
- 5) The TT&C receives any required central or digital timing signals from the CMS.
- 6) The TT&C provides processed and decoded command words to the CMS.
- 7) The TT&C receives stored spacecraft status and parametric data from the CMS to be formatted for transmission on the downlink.

c) RF "Uplink and Downlink" Path

This is the RF interface for receiving and transmitting data. Typically this is an interface with the ground system. However, nothing precludes interfaces with other locations (e.g., a satellite or the Shuttle).

d) Attitude, Translation, and Pointing Subsystem (ATPS)

- 1) For mechanically articulated antennas the TT&C provides "pointing" or location information to ATPS (via CMS). The ATPS directs the antenna articulation control and controls the spacecraft attitude to point the beam to the correct location.
- 2) For electronic steering or beam shaping of antennas, the TT&C receives from ATPS (via CMS) spacecraft position and attitude information. The TT&C then controls the beam(s) direction.
- 3) The TT&C may perform tracking of the uplink carrier (e.g., monopulse or conscan) to provide pointing information. This "signal location" information would be used by the TT&C to point the Multi-Beam Antennas (MBA's) and Electrically Steerable Antennas (ESA's). The TT&C would provide this data to the ATPS (via CMS) to point articulated antennas or for spacecraft attitude control.

e) Temperature Control Subsystem (TCS)

- 1) The TT&C provides temperature and status information to TCS via the CMS.
- 2) Some temperature control management will be performed within the TT&C. However, for inter-subsystem or system temperature effects, TCS shall direct the control.

f) TT&C Internal Interfaces

1) Service Function Control

The selection of operating modes and elements for the TT&C will be performed by the Control and Monitor (CM) functional element. This function will receive commands and execute stored commands and command sequences to perform the required TT&C service functions. This is the control for all TT&C functional elements.

2) Maintenance of the Service Function

The Control and Monitor functional element interfaces with the various internal TT&C sensors and the CMS for the information required to judge correct operation of the TT&C. This unit will modify the TT&C state, select redundant units, do whatever is necessary to meet performance requirements or go to back-up degraded performance modes.

The TT&C will internally decide how to maintain the service functions via control of its internal elements. In some cases the function will require another subsystem's action. For these cases the C&M provides the information to those subsystems via the CMS for action. Those subsystems will, of course, have to accept those "requests" on a preestablished (but modifiable) basis.

The sensors for judging performance could range from the common type telemetry sensor (e.g., AGC, Helix Current) to complex specialized sensors such as telemetry function test receivers.

5. Block Diagram

The functional block diagrams for the TT&C are shown in Figures D2-3, -4, -5, and -6.

B. EVALUATION

1. Benefits

Potential benefits that could be realized from use of the TT&C in a decentralized system design architecture are identified as follows:

- a) The subsystem is relatively independent of the spacecraft system and other subsystems.
- b) The functions and requirements on the TT&C can be verified through test at the subsystem level without complex system-level subsystem inter-reactions.
- c) There are fewer interfaces with other subsystems.
- d) Complex analog interfaces become simple digital interfaces.
- e) Functionally complex interfaces become simplified since, many complex system interactions and interfaces are now within the subsystem.
- f) The subsystem can be function alone without interference from or to other system elements.

- g) The subsystem hardware and software designs are somewhat self contained. The amount of system-level knowledge needed by the designers is reduced. However, the system design phase has to be very extensive to insure subsystem design consistency with system-level requirements.
- h) Internal TT&C decisions can be made on an internally controlled executive basis and probably more rapidly than in an over-taxed centralized system.
- i) The technology exists for relatively low power and moderate speed processing as well as for low quantity data storage (memory). The subsystem should not require large memories or have requirements for sizable computing power.
- j) The subsystem performance can be optimized at the subsystem level.

2. Performance Features

Significant performance features of the TT&C when used in a decentralized system design architecture are identified as follows:

- a) Performance optimization should be possible because of internal decisions.
- b) Individual analysis/decision/action times should be reduced with decentralized processing. There could be many parallel activities occurring simultaneously.

3. Flexibility

The TT&C could make significant internal design changes without influencing the system design, as long as the basic functional and interface requirements are essentially unchanged. Conversely, the spacecraft system design changes might not cause large changes in the TT&C depending on the nature of the change. This would be on a mission-to-mission basis. The decentralized system should be very efficient in making changes needed for various missions. In particular, using individual functional element hardware and tailored software should minimize the problem of adding or deleting functions or hardware. Basically, the subsystem would employ modular design techniques. Having hardware and software functions designed in a modular one-for-one basis should make changes easier to implement.

4. Constraints

Pertinent constraints levied on the TT&C when used in a decentralized system design architecture are identified as follows:

- a) PWR has to provide, via CMS, "available" power status information. The TT&C controls the power internally. However, the control decisions have to be made on the basis of the power quantity available for each standard and non-standard mission phase. PWR also will have to supply a power bus for critical functions such as command or memory.
- b) The TT&C and ATPS will have to carefully define the functional interface allocations. The TT&C needs to transmit and receive signals via the antennas to carry out its functions. To do this the spacecraft must maintain a specified attitude and/or keep the antennas pointed at the proper location. Also, the TT&C can locate the position of the transmitter sending signals to the spacecraft via RF tracking techniques (i.e., the TT&C can determine the spacecraft attitude).

The TT&C will supply its pointing requirements to ATPS and it will supply pointing information from the received signal to ATPS. The TT&C will provide control to antennas which are electronically steerable. For mechanically articulated antennas which can influence the spacecraft attitude, the TT&C will issue its pointing requirements to ATPS. The ATPS or ground control will have to resolve pointing request conflicts.

- c) The TT&C and TCS will have to supply information back and forth. The TT&C power amplifiers are typically big power and thermal dissipators.
- d) Access to CMS information from other subsystems or from the system will probably be on a time shared basis. Careful design will have to be accomplished to establish the appropriate or acceptable "sample" interval, especially for critical functions.

5. Tradeoffs

TBD

6. Other Drivers

TBD

7. Detractors

Some potentially negative attributes of the TT&C design architecture defined herein are discussed as follows:

- a) The TT&C will require significantly more internal design effort. This design will be in functional and hardware design as well as a significant effort in software development. The overall system design effort should be comparable to that required for a centralized system.

- b) The subsystem test time will increase significantly.
- c) The subsystem complexity will increase significantly.
- d) The subsystem costs will increase.
- e) The internal TT&C interfaces will be more complex.
- f) Subsystem functional and design requirements will be more complex.
- g) Self-checking fault tolerant computer technology is almost but not quite here (for reasonable mass, power and cost).

C. Conclusions

One of the key advantages of decentralized processing to the TT&C is having one set of design engineers do the functional and detailed design of a complex subsystem. This centralizes design responsibility to a fairly low level. This is opposed to having a central design, an ATPS design, a PWR design, etc., all of which interact significantly with one another and all of which are pretty much independent of one another in design details. The decentralized approach will require a very heavy front-end top-down system design to insure that the spacecraft design will meet mission requirements.

Subsystem performance validation at the subsystem level has many benefits. Expensive problems at the system level should be minimized. System type problems should be more readily identified and solved at the subsystem level.

There will be problems in how to functionally allocate and control some functions which have classically been handled by an executive. For example, the TT&C needs to increase the transmitter power and has to exceed its available raw power. How are the priority needs established and controlled? Another example would be the pointing of an antenna when other pointing requirements conflict. There are solutions to these types of intersystem functions in a decentralized system.

V. PAYLOAD DESIGN ARCHITECTURE

A. DESCRIPTION

The following is a very generalized description of a payload for a spacecraft architecture designed around decentralized processing. This generalized (mythical) payload could be one simple instrument, a complex communications subsystem like the one on DSCS III, or an even more complex subsystem.

The payload is basically the key reason for the existence of a spacecraft. It is the subsystem that is supported by the spacecraft bus service functions. Fundamentally the payload is placed on board the spacecraft to collect information and return this information to the ground. (Some payloads like those of a killer spacecraft probably fall outside the above definition. Also, the unique requirements of manned payloads are not included here).

1. Function

The payload is typically the subsystem which forms the basis for the mission. Most of the spacecraft service functions are directed at satisfying the needs of the payload. The function of the payload is to gather and return information. In general, most payloads have as key functions the collection of data or information via some instrument; the processing and storing of the information; the distribution of the processed or raw data; and, the control and monitoring required to carry out the payload functions.

The functional block diagram of the payload is shown in Figure D2-7.

2. Functional Elements

The following represents a partitioning of the payload subsystem into functional elements:

a) Instrument Function

The instrument function is usually made up of a device or a group of devices which collect desired information (e.g., T.V. camera, infrared spectrometer, R.F. receiver). The instrument can output digital and/or analog data to the data conditioning function.

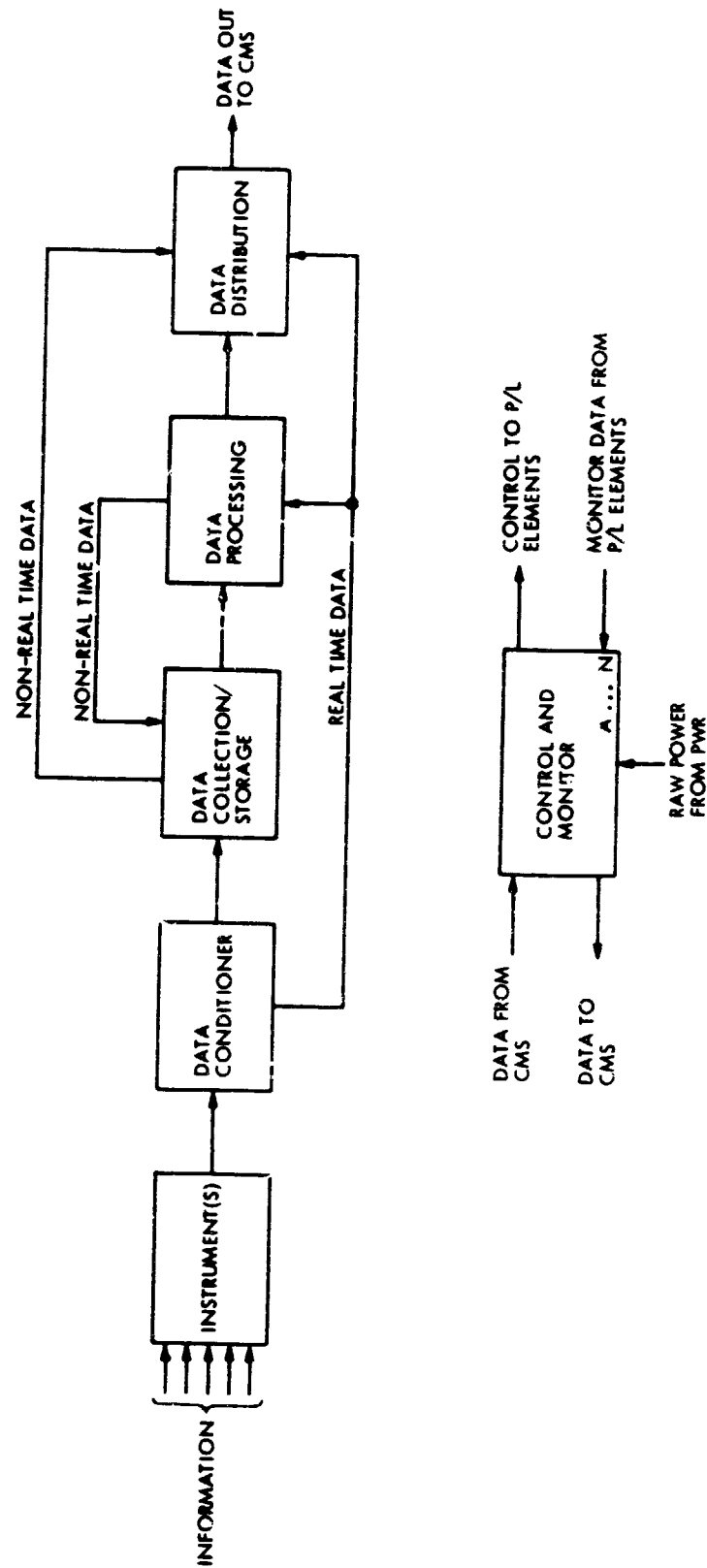


Figure D2-7. Generalized Payload (P/L) Functional Block Diagram

b) Data Conditioning Function

Typically, the information from the instrument requires some type of conditioning. This may be a simple buffering operation, an analog-to-digital conversion, or data compression. Basically, this function prepares or places the data in a format suitable for storage or transmission. For example, analog imaging data may be conditioned, converted to digital words, buffered and outputted to a storage system for non-real time data transmission. The data could also be sent directly for real time transmission via the CMS.

c) Data Collection/Storage Function

After the data has been converted to digital words it may be stored for later processing or transmission. This collection/storage function could be a simple buffer memory, a tape recorder, or a combination buffer/tape.

d) Data Processing Function

In a simple system, the data processing function could be passing data through for direct transmission.

In more complex systems, the processing function could code the data for improved error rates or it could perform extensive data compression operations. Basically, the processing is a simple or complex modification of the data to prepare it for eventual transmission on the downlink. This processing could be off line for stored data or in real time. Its output could be provided in real time or routed to the storage function for later transmission.

e) Data Distribution Function

This function takes the real or non-real time data and outputs it to the user via the CMS. The output could be a simple single digital interface or it could be multiple interfaces for multiple destinations (e.g., X-Band and UHF data transmission).

f) Control and Monitor Function

The control and monitor (C&M) function serves three basic functions. It receives the commands, stores the commands, and executes the commands required to carry out the payload data collection and return function. The C&M also provides the fault detection, isolation and correction functions required for the payload to perform its required functions. The C&M has the CMS as its primary interface. It sends data to the CMS and receives information from the CMS (e.g., the instrument sees a star of the right magnitude and needs to know the payload position or orientation to "name" or categorize the star; or, the payload instrument

is pointed wrong and the payload needs to redirect it via a spacecraft orientation change). Any sensors required for fault maintenance are part of the C&M functional element.

3. Functional Requirements

The functional requirements imposed upon the payload by the decentralized system design architecture are defined as follows:

- a) Collect data via its instruments. The data shall be appropriately processed and conditioned for subsequent return of the data to the user through the CMS.
- b) Provide to the CMS all status and information required by other subsystems.
- c) Receive via the CMS all information required to carry out its functions.
- d) Complete its required functions without direction from outside sources except for executive control commands from the ground mission control.
- e) Accommodate the reliability, redundancy and failure policies established by the project.

4. Interface Requirements

The interface requirements imposed upon the payload by the decentralized system design architecture are defined as follows:

- a) Power Subsystem (PWR)
 - 1) The payload will typically receive raw power and provide its own internal conversions to get the appropriate voltages.
 - 2) The payload controls the distribution of the input power internally (i.e., provides its own power management based on allowable load information received from PWR via the CMS).
 - 3) For critical loads, separate power interfaces will be provided.
- b) Common Memory Subsystem (CMS)
 - 1) The payload can request and receive preprogrammed stored sequences from the CMS memory (however, some or all of these could be stored in payload memory).
 - 2) Mission control executive commands will be received via the CMS.

- 3) All required payload status and information will be provided to the CMS.
 - 4) The payload data and information will be sent to the CMS for distribution (typically to the TT&C for downlink transmission).
 - 5) The CMS will provide the payload with all required information from other subsystems.
 - 6) All spacecraft digital timing will be received from the CMS.
- c) Temperature Control Subsystem (TCS)

The payload provides data through the CMS to the TCS for maintenance of prescribed temperatures. Some specialized temperature control is accomplished through internal payload temperature management. Those areas of temperature control which influence other subsystems are controlled by TCS.

5. Block Diagram

The functional block diagram for the payload subsystem is shown in Figure 7.

B. EVALUATION

1. Benefits

Potential benefits that could be realized from use of a payload in a decentralized system design architecture are identified as follows:

- a) The payload is frequently the subsystem about which a spacecraft bus is designed or redesigned. Having the payload as divorced as possible from complex subsystem interfaces via an appropriate decentralized processing design makes it more independent of the spacecraft system.
- b) The performance requirements are more easily validated at the subsystem level without simulating complex subsystem and system interfaces.
- c) There are fewer interfaces with the spacecraft.
- d) The information interfaces are simple digital interfaces.
- e) Functionally complex interfaces become simplified.
- f) The subsystem can function alone without significant interference to or from the other subsystems.

- g) The subsystem design is somewhat self-contained. If the system design is rigorously defined, the subsystem design can proceed somewhat independently.
- h) For complex payload subsystems, which typically have highly specialized functions, a separate control and monitor system is probably desirable. The subsystem designers should be able to do a better job of designing the subsystem; however, a simple payload could be overdesigned with its own processing capability.
- i) The response times should be faster in the self-contained processing approach.
- j) It should be easier for the subsystem engineers to optimize the performance with integrated processing capability.

2. Performance Features

Significant performance features of the payload when used in a decentralized system design architecture are identified as follows:

- a) Decision times and data processing capabilities will probably be faster than with a shared processor.
- b) There should be performance improvements due to design optimization.

3. Flexibility

The internal payload design could change significantly without making significant spacecraft system changes. This assumes that the functional and interface requirements remain the same. Conversely, other subsystems could change significantly without causing large changes to the payload. The self-contained payload approach is somewhat like the Goddard MMS approach, although they do not use enough distributed processing. Depending on payload complexity, a "standard" or building block bus could serve a wide variety of "smart" payloads. Many of the payload instruments are becoming "smart" to simplify spacecraft interfaces and to establish multimission applicability.

4. Constraints

For the most part, constraints will be dependent on the exact functions performed by the payload. However, a few possible constraints are identified as follows:

- a) Power will be received "raw". All conditioning, controlling, and monitoring of power will be accomplished by the payload. This control and monitor function will probably be based on information from power via CMS (e.g., mission phase or emergency power-down state).

- b) Frequently the payload instruments need precise pointing to acquire the information required. A set of interface functional requirements between ATPS and the payload would need to be well defined.
- c) The payload is frequently a major power consumer. A well defined thermal control interface between TCS and the payload will have to be established.
- d) The data to and from the payload will probably be on a time shared basis. This will require careful design to insure that the timing is right.

5. Tradeoffs

TBD

6. Other Drivers

TBD

7. Detractors

Some potentially negative attributes of the payload design architecture defined herein are discussed as follows:

- a) The payload will require more internal design effort than with a centralized system.
- b) Subsystem validation time could increase.
- c) The payload complexity could increase.
- d) The subsystem costs could increase.
- e) Self-checking fault tolerant computer technology is advancing; however, it could be costly in mass, dollars, and power when compared with current technology.
- f) Overall use of an internal processor may be inefficient (e.g., lots of memory and computational power could be idle most of the time, especially for maintenance and monitor functions).

8. Conclusions

The payload frequently governs the design of the spacecraft bus or service functions. The decoupling of the payload from the spacecraft bus by decentralized processing makes a spacecraft bus design less susceptible to large changes when the payload design changes.

Decentralized design also places design responsibility at a lower level. With mission and system requirements being well defined, the design responsibility can be delegated to the subsystem engineers.

The individual payload and the spacecraft system and subsystem performance can be validated relatively independent of one another. In the case of the typically complex payload, having independent validation of the spacecraft system could have significant cost and schedule advantages.

There will be problems in how the payload obtains needed services from the spacecraft bus. The functional and detailed performance requirements will have to be very carefully defined. An example would be how the payload would "request" pointing the spacecraft or instrument at a specific location.

VI. NAVIGATION SUBSYSTEM DESIGN ARCHITECTURE

A. DESCRIPTION

1. Function

The primary function of the Navigation Subsystem (NAV) is to maintain the spacecraft at its assigned longitude within a prescribed set of limits. In addition, the payload users require data describing the current and predicted spacecraft positions and velocities. In a decentralized system, NAV is also assigned certain mission control responsibilities. These latter responsibilities include 1) keeping track of mission phase and 2) controlling propellant consumption rates by appropriately varying the station-keeping boundaries.

2. Functional Elements

To accomplish its functions, NAV requires sensors to measure the orbit of the spacecraft and a set of computer algorithms in a self-checking fault tolerant computer. The algorithms transform the measurements into, first, an estimation of the current orbit and then into the maneuver commands required to control the orbit. The current strawman subsystem uses an Earth sensor, a star sensor, and a set of six body-mounted sun sensors to provide the required measurements.

3. Functional Requirements

The functional requirements imposed upon NAV by the decentralized system design architecture are defined as follows:

- a) Acquire Earth, Sun and star measurements
- b) Smooth, edit and calibrate measurements
- c) Estimate spacecraft orbit and related parameters
- d) Plan and command maneuvers required to maintain station
- e) Provide telemetry data to CMS
- f) Provide navigation parameters to CMS
- g) Provide self-maintained fault tolerance to all internal single-point failures
- h) Maintain absolute time reference

4. Interface Requirements

NAV interfaces directly with only the CMS and PWR. However, Navigation sensor data may be on a separate bus supplying data in parallel to both ATPS and NAV.

5. Block Diagram

The functional block diagram for NAV is shown in Figure D2-8.

8. EVALUATION

1. Benefits

The use of a decentralized architecture benefits NAV in several ways by simplifying the interfaces with other subsystems. In general, a number of different subsystems interface with NAV and by passing this data through a common buffer (the CMS) the details of the interface can be better described and controlled. This structure will automatically force a degree of consistency between individual interfaces. The structure also allows the optimization of the computer utilization. The NAV function is heavily computer dependent; however, the characteristics of the computational requirements are very different from ATPS, the other major computation center. ATPS is basically a "high" frequency subsystem where the updates and commands are recomputed on short time intervals. NAV, on the other hand, is a "low" frequency subsystem and performs massive computations on fairly large time intervals. The difference in the frequency characteristics of the two subsystems is expected to make the integration of these functions in a single centralized computer much more difficult than the integration in separate computers.

2. Performance Features

As noted previously, NAV benefits from a decentralized architecture by simplifying the interface requirements and by allowing for efficient usage of the computer resources. Especially in the latter case, the use of a separate computer limits the impact of changes in the NAV algorithms and should allow for more efficient development and testing.

3. Flexibility

The use of a separate computer facility provides NAV with the capability to adjust the real-time computational cycle, in response to current requirements, without impacting other subsystems. The subsystem is also better able to adjust to program updates and modifications.

4. Constraints

The potential weak link between the various subsystems, implicit in a decentralized architecture, will require detailed system-level design to insure that failures (or symptoms) involving two or more subsystems are handled properly. If not, longer time constants could result, increasing the potential for unstable iteration loops to exist between subsystems. Due to its coupling with a number of other subsystems, NAV is particularly susceptible to this problem.

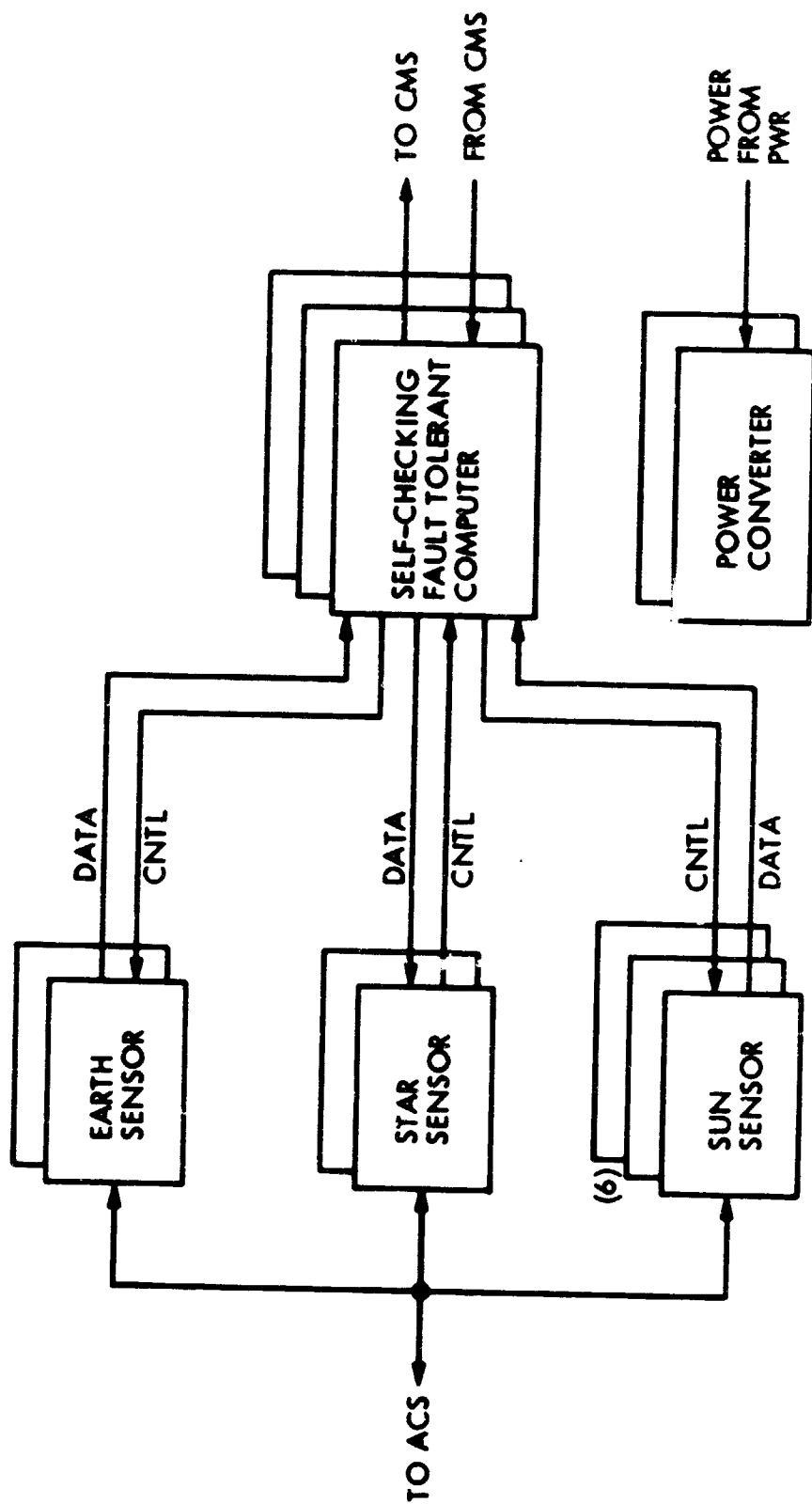


Figure D2-8. Navigation Subsystem Functional Block Diagram

5. Trade-Offs

Strong system-level design will be required to insure that the total set of system-level functions are properly and constantly assigned to the various subsystems. Since NAV is expected to be assigned a number of system functions (propellant control, for example), early definition is required.

6. Other Drivers

NAV may be influenced by the required intersubsystem data transfer rates. Minimizing these rates will benefit NAV.

7. Detractors

A decentralized architecture results in the assignment of additional functional responsibilities to NAV. The additional work will decrease the efficiency of the baseline navigation computations.

8. Conclusions

A decentralized system appears to be a very workable architecture with major benefits to navigation. Detailed analysis of intersubsystem communication rate requirements is needed before a final configuration can be chosen.

VII. ATPS DESIGN ARCHITECTURE

A. DESCRIPTION

The Attitude, Translation, and Pointing Subsystem (ATPS) control intelligence (logic) is partitioned into microprocessor modules located within associated devices which are responsible for local control and collection of data. An ATPS executive processor coordinates the handling of the various data sources and performs high-level tasks such as commanding, formatting TLM, data reduction and analysis, adaptive control, fault validation, reconfiguration control, mission mode equipment usage, and all high-level autonomous management.

The ATPS, Figure D2-9, consists of a network of distributed microprocessors connected by a simple internal bus system. Each device (sensor and actuators) assembly contains an embedded processor module with memory and an I/O interface to the Subsystem EXEC via the internal bus. Both intramodule operations and I/O functions are synchronized by a real time interrupt (RTI) with a period on the order of two msec. I/O operations are initiated only in the occurrence of a RTI interval thus avoiding time-critical design and the burden of controlling high-rate I/O operations.

1. Function

The ATPS provides the sensors, computation, and actuation required for real-time control of spacecraft attitude, translation, and payload or other instrument pointing. The ATPS functions in cooperation with other autonomous subsystems via the CMS to furnish requested data and services and to respond to requested actions from ground control. TLM is sent to the TT&C on a non-interference direct memory access (DMA) line via the CMS, and attitude sensor data is sent to NAV. Required delta-V thruster operation is commanded to satisfy action requests from NAV which specify thrust vector orientation and velocity increment.

2. Functional Elements

The ATPS consists of the following three major functional elements:

a) EXEC processor

The EXEC communicates with other subsystems via the CMS time domain multiplexed (TDM) spacecraft Intercom bus. The EXEC has data, command, and power links to all ATPS peripheral devices via an internal bus structure under software (S/W) control. The associated EXEC electronics accepts and conditions 28 Vdc power, and generates all required secondary levels. The power relay matrix for control of peripherals as well as EXEC modules resides in the EXEC electronics, and full access is available to the EXEC S/W and special hardware detection circuits for heartbeat and other fault monitors. The EXEC also accepts a master timing synch from the CMS, and distributes required

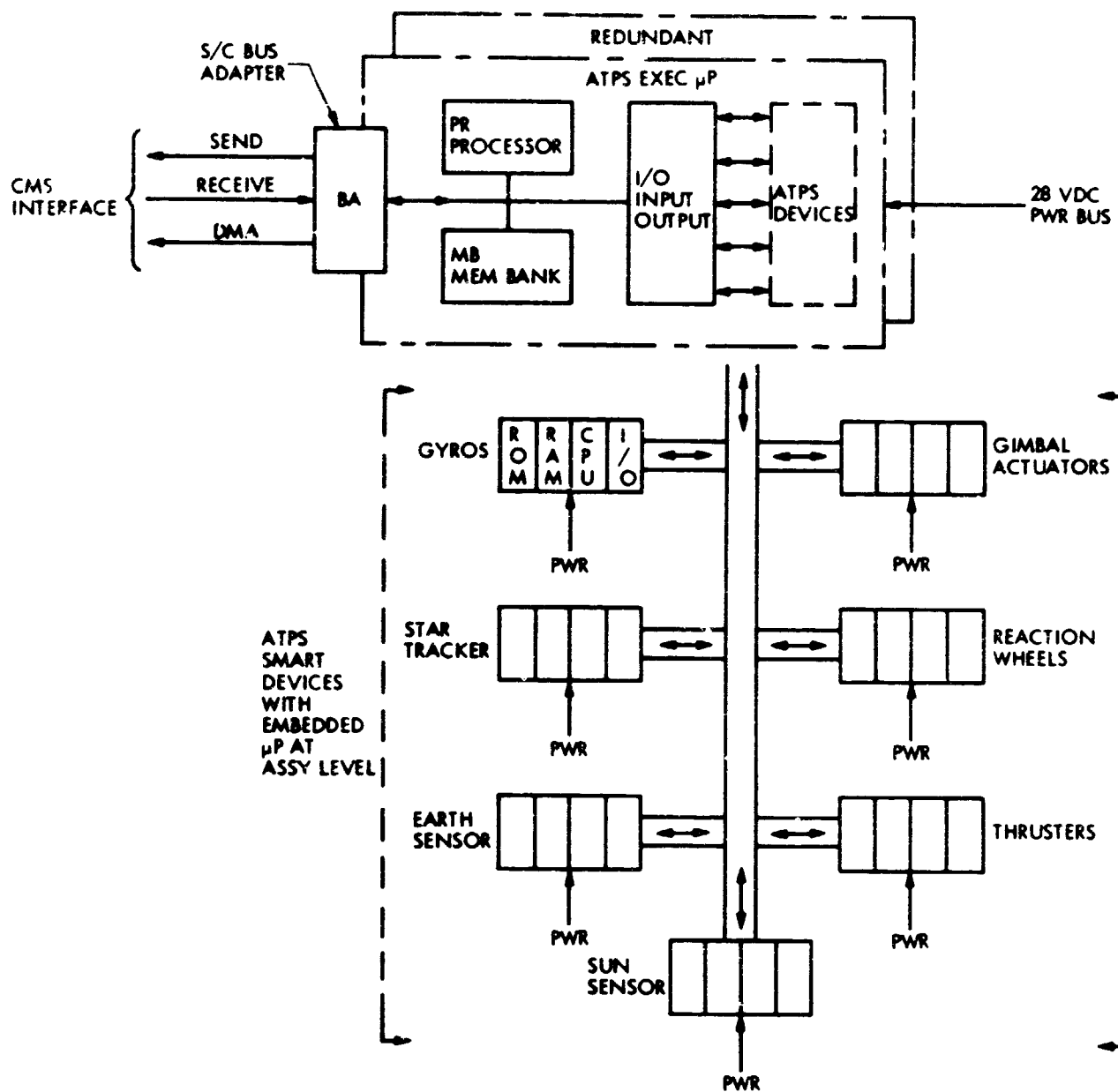


Figure D2-9. ATPS Functional Block Diagram

peripheral sync or clock signals. In general, the internal ATPS bus operates asynchronously via the RTI under S/W control, and provides for transient power outage protection. The EXEC processor contains ROM, RAM and non-volatile (NV) memory in the form of read/write bubble memory. The subsystem control laws and all state estimation is implemented in the firmware. Sensor post-processing and coordinate transformations; optical and inertial sensor data mixing and scaling; and proportional, nonlinear, and adaptive control loop commanding are carried out in the EXEC.

b) Sensors

Optical sensors function in two major spacecraft subsystems, the Auto NAV and ATPS. The representative sensor set utilizes three types of sensors: Earth, Sun, and Star reference sensors. The Earth sensor controls the spacecraft attitude in an earth reference frame and provides the direction from which angles to the sun and to the reference star are measured. The star tracker provides yaw data for the ATPS and latitude information for NAV orbit determination. At geosynchronous orbit, the use of Polaris as a reference star allows the latitude measurement to be relatively independent of the longitude determination. The data from the sun sensor together with the solar ephemeris and on-board clock time (orbit time) is used to compute longitude. The three sensor sets have separate control, interface, and calibration requirements under supervision of the ATPS EXEC. Data types are provided to NAV in accordance with requests and NAV can perform its own type of calibration using that data. Since the ATPS has data bandwidth needs far greater than NAV (real-time control vs extended-time orbit determination), the critical time-dependent control and fault protection functions for the sensors must be done by ATPS. A detailed discussion of each of the individual sensor types is provided as follows:

1) Solid State Star Tracker

High precision spacecraft stabilization and payload pointing usually depend on the use of a star tracker. Parameters such as field-of-view, sensitivity, acquisition logic and response time are typically fixed in star tracker hardware, but mission requirements often dictate a wide range of values. In most cases, each user has been required to design a unique star tracker for specific applications. This results in producing one of the most expensive devices on the spacecraft for each mission type. Most present trackers utilize an image dissector tube which has a high cost, limited life, and operates at high voltage.

The ATPS Solid State Star Tracker utilizes a charged coupled device (CCD) to replace the image dissector tube. The CCD is a detector that offers significant improvements in all parameters over the image dissector, e.g., longer life, higher reliability, lower cost, simplicity, high versatility, and low voltage operation. CCD's are manufactured with standard integrated circuit techniques by semiconductor companies. The CCD output signal format is serial discrete charge packets ideally suited to digital data processing. A microprocessor is used to perform the data readout/acquisition, formatting and output processing functions that were traditionally hardwired in place. Simple variations in programming can alter the basic tracker functions without necessity for hardware design changes. This provides a high degree of functional adaptability for multimission use.

The basic concept of this sensor consists of an optical system, a CCD area imager placed at the focus of the optics, a microprocessor to handle data acquisition, data processing/formatting/output control, and a control program stored in Read Only Memory (ROM). The CCD integrates a charge pattern resulting from the star field imaged upon it during the photon integration time. The charge pattern is then read out serially line by line to an analog-to-digital converter (ADC) which makes selected star data available to the microprocessor. The microprocessor then performs data editing, analysis, and error signal generation in accordance with the stored program and presents the data to the I/O interface registers. The decision logic and computational algorithms in the processor/program can be adjusted to the mission requirements, thus relieving the subsystem EXEC from having to perform these tasks. The star tracker becomes, in a sense, an intelligent peripheral in a computer-based autonomous control system. It employs adaptive strategies for 1) star mapping, 2) initial search and acquisition, and 3) tracking on single or multiple stars for spacecraft attitude.

2) Digital Sun Sensor

The ATPS Fine Digital Sun Sensor device consists of two subassemblies, a sensor head and an electronics unit with an embedded microprocessor. The sensor head contains the optical elements for sensing the sun angle in two axes. The electronics unit processes the sensor-head signals and outputs a binary number that is used in the sensor transfer function to solve for the sun angle. The transfer function contains terms which are a function of the output binary number. The coefficients of the transfer function are set during testing and calibration, and the complexity of the transfer function depends on the desired measurement accuracy.

Signal processing for the fine sun sensors can be performed using a variety of techniques that can be tailored to particular mission requirements. Typically, the raw photocell data is amplified, filtered and combined with the other cell's data, multiplexed into an ADC, and presented to the microprocessor. The processor performs the logic to resolve ambiguity when combining the fine and coarse sensor data, and calculates the sun angle from the transfer function algorithm. Data output is in binary format which may be all natural binary or a combination of Gray code and binary depending on the processing requirements. Serial or parallel digital formats are available in buffered form, and other interfacing specifics for data output timing can be adapted to the particular mission requirements.

3) Earth Sensor

The Earth Sensor operates by balancing the infrared (IR) radiation from the earth on pairs of solid-state detectors. It provides spacecraft pitch and roll error signals for stabilization of the yaw axis to the local vertical. The array of detectors within the sensor detects the radiation level change between the earth's atmosphere and the spatial background. Opposing detector elements are paired and when the spacecraft deviates from the Nadir alignment, a null error in two axes is generated. Should the sun intrude on any one detector the signal from the opposite of the pair is automatically switched out, and the remaining detector pairs are selected for the two axis error computations. Control choice can remove one or more detector pairs from use whether or not sun intrusion exists. The sensor also provides an earth presence signal for use in initial acquisition/reacquire maneuvers. All signal processing and detector fault protection is under the embedded microprocessor control. An I/O module provides interfacing to the ATPS EXEC via the internal subsystem bus. The detector signal switching logic, signal mixing and axis error computations, sun and earth presence logic, and calibration/self-test routines are stored in ROM program. This allows application flexibility and adjustment via software control.

4) Inertial Sensors

ATPS inertial sensors (gyros) function in several ways, depending on the system design, to provide an independent control path for spacecraft attitude and pointing stability. Modern high precision rate-integrating gyros have post-calibration drifts and noise errors so small that they provide a superior sensor for high-bandwidth maneuvers and payload pointing compared to optical sensors. The optimum systems usually combine both low-bandwidth optical sensors and gyros in an aided-inertial mode with state estimation filters. The optical sensor provides the long-term position reference update to gyro random drift, and the shorter term

position and rate data is taken from the gyro between updates. Inertial control for acquisition and search maneuvers, autonomous recovery from loss of optical references, preprogrammed nadir pointing or offset nadir scanning profiles, autopilot/thrust vector control during delta-V station or orbit burns, and other dynamic control tasks are typically designed around the inertial sensor capability. With the advent of high precision tuned-rotor gyro technology, and the even newer fiber-optics Sagnac Effect rotation sensor developments, the continuous long-mission use of the optimized aided-inertial control technique is feasible.

The ATPS embedded microprocessor inertial sensor assembly will provide the capabilities for performing 1) rate and position computations and 2) digital transfer of motion state data rather than raw increments of rate/position. Adaptability to mission and environments is possible by programmable drift compensation and noise filters, frequency response shaping, signal mixing and scaling, temperature compensation, test and calibration interfaces, and mode control logic. Lower communication rates to the ATPS EXEC via the subsystem internal bus is a benefit of preprocessing raw signals, and performing device unique health checks at the local peripheral point of control.

- c) The ATPS utilizes three types of control actuators to provide orientation of the spacecraft and gimballed payloads. Reaction wheels provide precision/proportional control of attitude; thrusters (mono or bi-propellant) provide subsystem translation and wheel unloading; and direct drive torque motors point the payload(s) in two gimballed axes. It is necessary for the ATPS to have control authority over all these devices since the basic stabilization function must account for all reactions (angular and linear momentum exchanges) and impulsive disturbances. The high bandwidth real-time control function of ATPS requires a highly integrated approach to actuator implementation, the control laws, and command generation of the subsystem EXEC processor. The ATPS also provides services to other lower bandwidth/less dynamic subsystems such as NAV and RFS.

The three types of control actuators are described in more detail as follows:

1) Magnetic Bearing Reaction Wheels (MBRW)

While attitude control using reaction wheels is an established technique, the increasingly severe demands of long mission life and high reliability with autonomous fault protection are strong concerns. For this reason, development of advanced technology wheels has been pursued with successful designs of internally redundant and non-contacting wheel bearings and motors. Active

and passive magnetic support loops under servo control, and dual segmented motor construction provides the equivalent of six reaction wheels in only three units. Electronic elements are substituted for mechanical bearings and motor brushes, bringing the prediction of reliability under control. Redundant circuits now allow the two-in-one concept to be realized along with higher precision of control due to the absence of bearing friction. Lower power is afforded by the use of efficient dc torque motors using electronic commutation.

A typical MBRW assembly consists of three orthogonally mounted wheels, each developing bi-directional torques in response to commands from the ATPS EXEC. In prior traditional control systems a degradation in performance has been experienced with fixed servo compensator parameters, i.e., the control filter constants may not be true over the entire speed range, and this also drives D/A converter resolution requirements. Conventional wheels are not readily adaptable to multimission needs or changing conditions of extended-time missions. The MBRW is a sample-data system utilizing an embedded microprocessor for adaptive software compensation and control. Modular partitioning of the MBRW functions allows high applications flexibility. These functions are: I/O circuits for digital interfacing with the ATPS internal bus; wheel speed controller logic and servo; wheel motor drive modulation logic/circuits; tachometer sensor logic; magnetic bearing servo logic and proximeter sensor circuits; and power supplies. These features under device software control provide the benefits of 1) very long life under continuous usage, 2) high precision attitude stability and accuracy, and 3) greater fault protection in autonomous operation.

2) Direct Drive Brushless Torque Motors

Gimballed payload articulation/pointing is provided by power efficient and accurate torque motors under proportional control by the ATPS EXEC. The absence of gearing friction and backlash and the smooth control of the payload line-of-sight for pointing and tracking are clear attributes of modern direct drive, electronically commutated, servo motors. Internally redundant motor windings, commutator sensors, and motor controls provide fault protection under microprocessor supervision. The embedded ROM software and I/O modules allow local monitoring of command response in the gimbal motion closed loops through resolver or shaft encoder feedback. The high bandwidth gimbal servos can function to carry out the ATPS macro-commands which are stored in the local RAM until executed by the gimbal servos under ROM control laws. Thus the EXEC to Actuator communication bandwidth need only be low to moderate, and the EXEC

is unburdened from the high data rate I/O operation otherwise required. Local self-test and calibration routines are stored in the ROM subject to supervisory validation control by the ATPS EXEC. Adaptability to a variety of mission payloads and types of articulation profiles (mosaic, raster, box, single or two axis motion, etc.) is provided by the microprocessor at the device level without reflecting changes to the subsystem design/EXEC.

3) ATPS Thrusters

The integration of detailed thruster and propellant supply control functions (such as catalyst-bed heater and temperature control, propellant solenoid and latch valve operation, chamber-pressure sensor data acquisition and scaling, solenoid current monitoring, and status of redundant devices) is feasible with an embedded microprocessor in the propulsion assembly. Control of all thrusters and supply valves, control of temperature, pressure, and flow of fluid level sensors, power-switching off the ATPS bus, and a host of dedicated local-management tasks can be done in response to higher-level commands from the ATPS EXEC. Testing of these functions is made more complete and independent of other assemblies. The command interface with the EXEC is readily emulated, and I/O data transmission via the ATPS bus structure is easily validated. Device unique self-test or 'health checks' can be performed at the local point and status flags sent to the EXEC. The removal of many housekeeping tasks from the higher level processor affords a more efficient use of EXEC computational and system management capabilities, and reduces the throughput demands on the total ATPS.

3. Functional Requirements

The ATPS shall satisfy the following functional requirements in cooperation with the other spacecraft autonomous subsystems and with spacecraft communications available via the Common Memory Subsystem (CMS):

- a) Provide autonomous attitude and pointing reference acquisition and re-acquire in all mission modes/phases.
- b) Perform autonomous attitude, translation, and payload pointing real-time control in all required mission phases.
- c) Perform attitude determination computations and analysis to support NAV and payload pointing functions.
- d) Perform its tasks in cooperation with the other spacecraft subsystems via the CMS communication link.

- e) Maintain integrity of shared sensors and other controlled devices such as actuators.
- f) Provide autonomous calibration of its devices in a manner transparent to other subsystems and users.
- g) Perform autonomous real-time propellant control and fault management.
- h) Respond to NAV requests for delta velocity thrust vector control options and thrust magnitude/velocity increment.
- i) Perform autonomous fault detection, isolation, correction/recovery of subsystem elements in a manner to maximize the user/payload operational effectiveness.
- j) Perform autonomous in-flight test and validation of subsystem elements and functions.

4. Interface Requirements

The interface requirements imposed upon the ATPS by the decentralized system design architecture are defined as follows:

- a) Communication by message words to and from all other spacecraft subsystems shall be via the Common Memory Subsystem (CMS) Intercom data bus.
- b) Telemetry data shall be sent to the CMS via a spacecraft direct memory access (DMA) bus. The CMS shall provide the TLM data to the TT&C via a dedicated output link.
- c) ATPS internal communications to all devices/peripherals shall be via a subsystem bus separate from the CMS communication links.
- d) The ATPS EXEC Computer receives unregulated spacecraft power (e.g., 28 Vdc) and converts this supply for internal use and distribution to all subsystem peripherals. The ATPS device power relay matrix is under control of the EXEC.
- e) The spacecraft master timing reference is provided via the CMS TDM Intercom bus to the ATPS EXEC.

5. Block Diagram

A conceptual functional block diagram of the ATPS with EXEC Computer and distributed processor peripherals is shown in Figure D2-9.

B. EVALUATION

1. Benefits

Potential benefits that could be realized from use of the ATPS in a decentralized system design architecture are identified as follows:

- a) The ATPS architecture provides all digital interfacing to the spacecraft level, and to its distributed embedded processor sensors and actuators.
- b) All interfaces are simplified and made to operate at a lower bandwidth since raw data is not transmitted between any computers/devices.
- c) Interfacing commonality and I/O standardization provides modular, non-interactive changes to the subsystem architecture. This allows multimission adaptability.
- d) ATPS EXEC Computer control over its own power relays and simple internal bus structure for data and power management allows efficient redundancy and expandability management.
- e) ATPS autonomy is considerably enhanced with local device health checks and higher-level EXEC performance/fault validation. Subsystem self-test is made transparent to the outside interfaces by distributed intelligence.
- f) Computer workloads are optimized for the specific functions allocated so that local embedded computers unburden the ATPS EXEC from routine/simple and device unique tasks. Capability of the EXEC is thus maximized and software is more efficient.
- g) Advanced technology sensors and actuators with internal redundancy provide savings in weight and cost while improving reliability and autonomy control.
- h) Subsystem fault protection and fault tolerance are greatly improved by the combination of all the attributes stated above. The effect is synergistic.

2. Performance Features

Significant performance features of the ATPS when used in a decentralized system design architecture are identified as follows:

- a) Intelligent peripheral devices are supervised and coordinated by a subsystem EXEC computer.
- b) Control system organization and performance is highly robust and readily optimized for both pre-launch and in-flight modes.

- c) Fault protection is placed at the point of action and constructed in layers to trap errors before they propagate through the system.
- d) Sensors and actuator peripherals represent advanced technology with enhanced performance by virtue of embedding special purpose processing and I/O.
- e) Devices have internal redundancy inherent in the designs. Block replication is not required unless dictated by a double failure protection policy for any device.
- f) Long-life (over 10 years) is feasible with computer and device technologies of the ATPS (solid-state optical sensors, non-contacting bearings and commutation, unfloated gyros/fiber-optic solid-state gyro, and all digital interfaces).

3. Flexibility

Characteristics of the ATPS related to its flexibility are described as follows:

- a) The distributed computer architecture and modularity provided by partitioned functions between the EXEC and peripherals creates a high degree of mission-to-mission flexibility for hardware and software changes.
- b) Complementing the subsystem internal structure, the CMS decentralized approach to spacecraft level communications and data handling provides a maximum degree of insensitivity to changes made at the system level.
- c) Adaptability to changing mission requirements during the build phase of the spacecraft system, as well as policy variations for ground-space segment operations, is made less painful with the distributed/decentralized design since major elements are decoupled from impact to a high degree.

4. Constraints

Pertinent constraints applicable to the ATPS when used in a decentralized system design architecture are identified as follows:

- a) Autonomous subsystems working in a cooperative mode will require strong emphasis on self-dependence that is not contradictory or in conflict with spacecraft resources and imperatives of the mission.
- b) Implicit in the design of the various subsystems will need to be a thoroughly thought-out software control over priorities/protocols which satisfies the mission goals and accounts for all the critical anomalies and contingencies. These scenario type situation responses must be validated in system design and properly placed in the on-board software of each subsystem.

- c) Computer hardware and software will need to be selected by criteria which account for the various subsystem processing differences while preventing a proliferation of generically different machines and languages. System level trades are clearly required, and the constraints must be strong.
- d) Subsystem responsibilities and tasks which are mutually supportive will need assignment based on criteria such as criticality of control, bandwidth of control function, interactive/disturbance nature of the response, and location of data required for the function.
- e) Decisions which affect on-board data rates between subsystems must be guided by the need to maintain low to moderate I/O transfers in order to obtain the full benefits of the decentralized system.
- f) Power conversions and internal distribution will be provided at the subsystem level.
- g) Transient power outage protection will be provided at the subsystem level.
- h) Communications on the CMS bus will be at the time slot allocated and should be compatible at no higher than a 2 ms repetition rate.
- i) Telemetry data will be routed to the CMS by a system DMA line so as to remove that throughput burden from the on-board subsystem-to-subsystem message management.

5. Tradeoffs

Local device unique health checks are intended to be in the memory (ROM) of a microprocessor dedicated to the operation of devices in a single assembly. Such a microprocessor would be a part of a distributed processing network comprising the ATPS. The devices and electronics in a given assembly could be managed by this microprocessor and the device health check algorithm used to remedy faults in these devices. A switch of block or functionally redundant devices would require that a microprocessor in any assembly is linked to a subsystem Executive processor through which such an action may be taken. The health of this link must also be monitored, perhaps like the monitoring done of the link between the AACS and CCS in Voyager.

The virtue of distributing the health check of a given device to the level of an individual hardware assembly is that the state of the device is transparent to any user. Any service using the device either receives a signal or establishes a degraded mode of operation as appropriate. The difficulty with such a distributed approach may be in the use of alternate devices to provide comparison test health checks. For example, in the Voyager TCAPU routine an important test which would detect thruster open anomalies made use of the output of the sun sensors. If the TCAPU

algorithm was located in the memory of a microprocessor in a separate propulsion subsystem, the sun sensor signals would need to be transmitted to that system. Given the critical nature of the test, this transmission must take place in a timely fashion. A fault in the transmission link or some delay in the receipt of the sun sensor signals may have a serious impact on the spacecraft. This is a good reason to re-think conventional system partitioning. In summary, the issue of distributing health checks to individual assemblies in the ATPS or subsystems throughout the spacecraft can only be dealt with in the context of the appropriate system and subsystem architecture of an individual spacecraft. Only a top-down design approach in a specific application can yield the information necessary to decide the degree of distributing the fault protection processing as opposed to maintaining a centralized location for such processing. The most practical designs may likely have a combination of both executive and distributed fault protection.

6. Other Drivers

Some ATPS design-related drivers that could significantly affect the performance and implementation characteristics of the ATPS when used in a decentralized system design architecture are defined as follows:

- a) The long range interests of spacecraft autonomous capabilities suggest that an inherent growth potential should be incorporated in the system and subsystem architectures. Providing for an expandable complexity of functional tasks without a comparable increase in equipment complexity and cost is a clear driver implicit in the evolutionary path to higher autonomy levels. The decentralized system with distributed computer subsystems affords the natural solution through programmable equipment in a modular architecture.

7. Conclusions

A distributed microprocessor-based ATPS design, applicable to a decentralized system design architecture, is an advanced approach in which the processing and control function is partitioned into a number of simpler, understandable and more adaptable elements. Complexity is removed from the level at which commanding and correlated verifications are concentrated, and placed in subsystem device interfaces which perform simpler dedicated tasks. In addition, the interfaces for system specification and design, testing, integration, and mission operations are significantly simplified.

Design and software complexity of the ATPS EXEC processor is reduced since device-unique logic and computation functions are carried out where practical at the local device point. The significance is that by giving the peripheral devices limited 'intelligence', their cost to multimission programs is actually

reduced. Standardized interfaces and simplified design and testing of these devices becomes possible. The added processor hardware and software for the improved peripherals is offset by the reduced burden on the EXEC processor electronics and interfaces. The device processor modules are not used in the general sense as "computers", though employing memory, CPU, and I/O elements. Sensors and actuators have a very specific and limited function; and, most device software is expected to be implemented in ROM with very limited capability for program modification in-flight using RAM patching in the small scratch-pad space. Mission flexibility of these 'smart' devices is best achieved by specifying process parameters under software ROM control (sampling-rates, scale factors, bias compensation, gains vs. operational mode, filter time-constants, etc.).

The modularity employed in the ATPS design results in a subsystem which is easily adaptable to multiple missions/requirements. Due to the separation of low and high level functions, and the all digital bus, addition or deletion of devices has little affect upon the rest of the system. This property also allows the addition of redundant elements to meet various reliability needs. This approach enables a sequential build-up of tested subsystem elements into a simplified system integration/test. The separation of software functions into smaller understandable parts, and a design which protects against local faults/degradation should result in savings to software development and mission implementation costs.

VIII. TCS DESIGN ARCHITECTURE

A. DESCRIPTION

1. Function

The function of the Temperature Control Subsystem (TCS) is to maintain the interfaces of spacecraft components and subsystems within specified temperature limits and defined heat rejection requirements.

In performing this function, the TCS is responsible for the design, sizing, and configuration of the entire spacecraft heat rejection system. This system includes radiators to reject energy to space; heat pipes, louvers, and fluid loops for energy transfer and rejection modulation capability; and heaters, louvers, and radiators in special areas where physical integration into the TCS is not practical (i.e., solar arrays, boom-mounted equipment, etc.).

2. Functional Elements

The primary functional elements of the TCS are defined as follows:

a) Temperature Control Computer Unit (TCCU)

The central control component of the TCS is the TCCU which is a fault tolerant computer that controls all functions of the TCS. This unit 1) receives, digitizes and evaluates TCS sensor data, 2) transmits digitized TCS sensor data to the CMS, 3) receives and evaluates other data from the CMS regarding the status of spacecraft components and subsystems, and 4) controls the operation/survival function of the spacecraft heat rejection system.

b) Power Conditioning Unit (PCU)

The PCU 1) conditions unregulated power provided by PWR as necessary and 2) distributes the resultant regulated voltages to the various components of the TCS as required.

c) Control and Switching Unit (CSU)

The CSU controls the heaters that maintain the component and subsystem interfaces within allowable temperature limits. Furthermore, the CSU, on command from the TCCU, switches the control mode (operation/survival) of the heaters.

d) Sensor Interface Unit (SIU)

The SIU receives sensor data from the TCS components and transfers it to the TCCU for digitization and transfer to the CMS.

3. Functional Requirements

The functional requirements imposed upon the TCS by the decentralized system design architecture are defined as follows:

- a) Monitor the temperature of the TCS components.
- b) Control TCS component temperatures.
- c) Monitor spacecraft system condition (operation/survival).
- d) Condition PWR input for use by TCS components.
- e) Control spacecraft system heat rejection components.

4. Interface Requirements

The TCS interfaces 1) directly with the CMS and PWR, and 2) indirectly with all subsystems through the CMS for the purpose of monitoring their condition. These interfaces are described as follows:

a) CMS Interface

The CMS is the only data communication link to the other components and subsystems of the spacecraft. The TCS digitizes the TCS sensor data and sends the data to the CMS. This data includes the TCS temperatures, spacecraft system radiator temperatures, heater modes, and TCS equipment status. The TCS also receives, monitors, and evaluates data from other spacecraft components and subsystems (via the CMS). It then uses this data to determine 1) spacecraft orbital conditions (i.e., attitude) and 2) operational conditions of other subsystems.

b) PWR Interface

PWR provides unregulated power to the TCS. Furthermore, PWR subsystem status information is transmitted to the TCS so that TCS spacecraft system-level control functions reflect this status. This data can be directly received from PWR or from the CMS.

5. Block Diagram

The functional block diagram for TCS is shown in Figure D2-10.

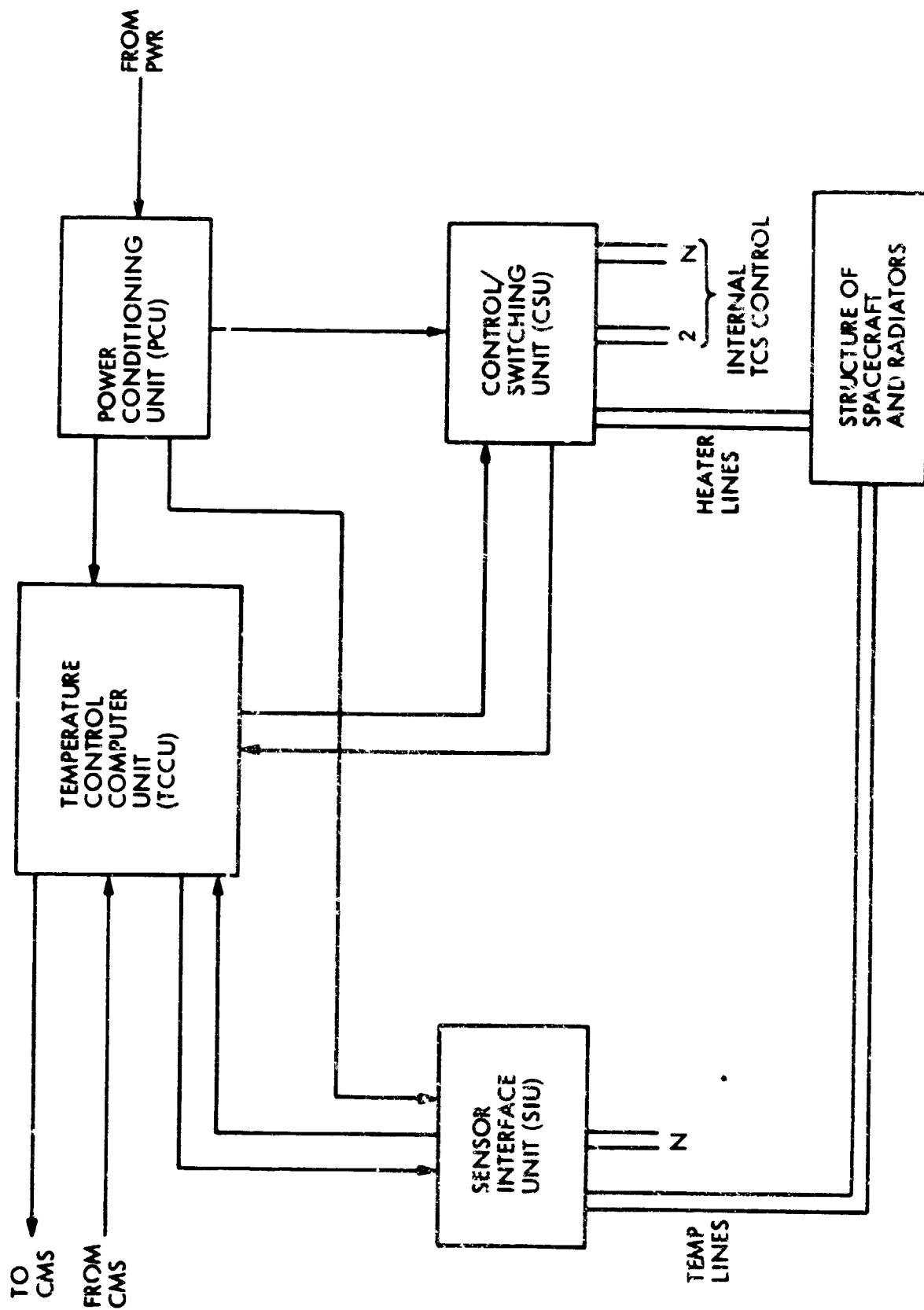


Figure D2-10. TCS Functional Block Diagram

B. EVALUATION

1. Benefits

The TCS controls only the spacecraft component and subsystem interface temperatures. The detailed temperature control of specific areas or components within a subsystem is done by the designated subsystem. This enables the subsystem to exercise direct control for more rapid response to the subsystem changes in condition. Furthermore, local control of specific areas minimizes the subsystem interface requirements.

2. Performance Features

The TCS for a decentralized system design architecture represents a comparatively simple subsystem. It is functionally limited to the temperature control of subsystem interface and spacecraft system-level heat rejection. Since subsystem interface requirements are minimized, system and component testing is simplified. Furthermore, the TCS can easily accommodate subsystem changes if the modified subsystem has the same power and temperature interface characteristics.

3. Flexibility

The TCS architecture is tolerant of a wide variation in mission parameters. Furthermore, other subsystems can be changed or modified without any effect on the TCS provided that the interface integrity is maintained. Another feature of the TCS is that its design can be modified to add capability without impacting existing spacecraft components or subsystems.

4. Constraints

The only major constraint levied on the TCS is the total heat rejection capability of the spacecraft (this is probably independent of computer processing architecture). The spacecraft heat rejection is defined by the configuration, size, and location of radiators, which is a function of the spacecraft configuration and launch vehicle allowable volume envelope.

5. Tradeoffs

The major tradeoff parameters associated with use of the TCS in a decentralized system design architecture are TCS simplicity versus control capability and system versus subsystem temperature control during failure condition determination.

6. Other Drivers

The basic driver for the design of the TCS in the decentralized system design architecture is the requirement to maintain its own fault tolerance. Other limiting drivers are the system electrical power availability and the spacecraft configuration.

7. Detractors

The TCS as used in the decentralized system design architecture provides no central control of different subsystem components. Each subsystem controls its own internal heaters. This could potentially cause problems for PWR if it cannot analyze power trends or requirements.

8. Conclusions

A decentralized system design architecture is potentially attractive for the TCS. The TCS design becomes very simple since each subsystem performs its own internal temperature control. Furthermore, subsystem interface requirements are minimized. These attributes reflect reduced design, test, and validation times with commensurate reductions in mission cost. Although a thorough system design effort is required because of the lack of a central control, the overall decentralized concept appears fully workable and very promising.

IX. POWER SUBSYSTEM DESIGN ARCHITECTURE

A. DESCRIPTION

1. Function

The primary function of the Power Subsystem (PWR) is to provide power to all other subsystems of the spacecraft commensurate with their needs. To accomplish this function, PWR performs energy acquisition and storage, power conditioning, and power distribution.

2. Functional Elements

PWR consists of the following functional elements:

- a) Solar Arrays
- b) Battery and Charger
- c) Power Chain
- d) Power Distribution Unit
- e) Power Computer

3. Functional Requirements

PWR is required to supply the spacecraft with uninterrupted power for the duration of the mission. During periods of eclipse, energy is transferred from the batteries (which must be charged during sungate activity) to the users. PWR must also supply operational status and telemetry information to the CMS. PWR must be internally fault tolerant and must be capable of power management during eclipse and subsystem failure.

4. Interface Requirements

The interfacing consists of digital signal lines to and from the CMS. Also, power output lines to users are required.

5. Block Diagram

A functional block diagram for a candidate PWR design architecture is given in Figure D2-11. Referring to Figure D2-11, the power sources consist of the solar arrays and batteries. They perform the functions of energy acquisition and storage, respectively. The power chain block of Figure D2-11 performs power conditioning while the power distribution unit distributes the conditioned power to the users. All elements are monitored and controlled by the power computer. Interface with the CMS for supplying operational status and telemetry information is also accommodated by the power computer.

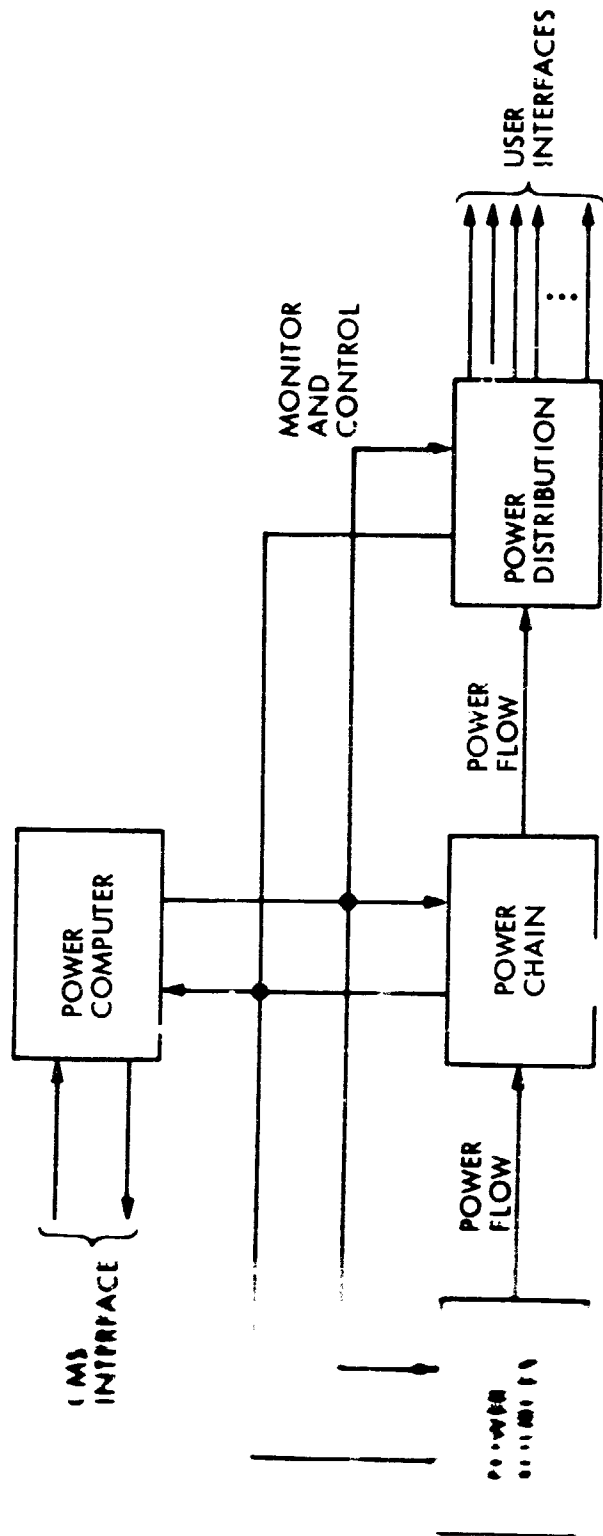


Figure D2-11. PDR Functional Block Diagram

B. EVALUATION

1. Benefits

A decentralized processing system has several benefits from the PWR standpoint. Reaction time to load failures is reduced because the power computer (PC) is continuously concerned with power functions. Reactions to load or PWR failures are made more efficient for the same reason. System validation is simplified by having more extensive testing at the subsystem level.

2. Performance Features

PWR operating efficiency is increased due to the continuously running PC. Emergency situations can be thoroughly analyzed without reducing the operational efficiency of the spacecraft. Because the PC is a dedicated unit, data rates may be lower, thereby reducing the bit error rate as well as susceptibility to electromagnetic interface (EMI).

3. Flexibility

PWR operates semi-independently of the other subsystems. Degradation in other subsystems can be made to have little impact on the PWR operational efficiency.

4. Constraints

Since no central computer is available to make decisions regarding power distribution, PWR must manage system power according to predefined (but mission-phase alterable) rules. In order to keep the system truly distributed, subsystems must internally manage the power allotted to them. Likewise, predetermined rules must have ATPS keeping the solar array sun oriented.

5. Tradeoffs

In a decentralized system, the system complexity is decreased at the expense of an increase in subsystem complexity.

6. Other Drivers

TBD

7. Detractors

The complexity of PWR must increase significantly to accommodate the system mission and its changing requirements. In order to maintain a high level of system performance, the PC must have a high level of system performance.

8. Conclusions

A point in spacecraft complexity will soon be reached where it is no longer feasible to have a central processor governing all subsystems. With today's technology and a good systems design team, it is possible, and indeed desirable, to incorporate a decentralized architecture onboard a highly sophisticated spacecraft for the following reasons:

- a) Reaction time to problems is reduced.
- b) Required computation speed is reduced.
 - 1) Cheaper processors
 - 2) Reduced noise immunity requirements
 - 3) Higher reliability machine
- c) System validation and test costs are reduced.
- d) System and subsystem operating efficiency is increased.

X. TEST AND VALIDATION METHODOLOGY

A. DESCRIPTION

The test and validation portion of a space system consists of procedures, and equipment to support those procedures, which are used to establish the ability of the system to meet its mission objectives.

1. Function

The function of test and validation procedures is to determine that the space segment, and the ground segment as appropriate, meet operational design requirements in both normal and abnormal environments, as defined; and, that fault survival capabilities function as specified.

2. Functional Elements

Functional elements of test and validation procedures for a decentralized space segment system include the following items:

- a) Establish proper operation of individual subsystems prior to integration.
- b) Establish proper conjunctive operation of these subsystems as an integrated system.
- c) Establish the health of redundant elements of subsystems and the provisions for autonomous management of these elements.

3. Functional Requirements

Functional requirements for test and validation procedures applicable to a decentralized space segment architecture are defined as follows:

- a) The ability to test individual subsystems, in conjunction with appropriate ancillary equipment, for compliance with design requirements.
- b) The ability to exercise autonomous redundancy provisions within individual subsystems and verify the health of those provisions, making use of ancillary equipment as required.
- c) Functions (a) and (b) for the integrated system of unique subsystems, with particular attention to subsystem interactions, using initially, the subsystem ancillary equipment and finally the integrated space and ground segments only, in their operational configuration.

4. Interface Requirements

Interface requirements for test and validation of separate subsystems of a decentralized system are defined as follows:

- a) Specified normal I/O interfaces with a simulated Common Memory Subsystem (CMS) within the ancillary equipment supporting that particular subsystem.
- b) Such special additional monitor and control access as may be required by that subsystem for simulation of conditions affecting autonomous redundancy management actions and monitoring the effects of such actions.
- c) Additional access to the interior of that subsystem as may be necessary to observe internal operation to the extent required to verify the normal CMS interface activity.
- d) Special access to indications of the health and status of static, invisible redundancy for those cases where this implementation is employed.

The interfaces specified, other than the normal CMS I/O, must continue to be available when the subsystems are integrated into the space segment decentralized system. This is necessary to permit observation and evaluation of system level interactions among the various subsystems as well as system level autonomous redundancy management.

The existence and use of these interfaces during system level testing must not affect system operation, with or without the ground segment active.

5. Block Diagram

Not applicable.

B. EVALUATION

1. Benefits

A decentralized system architecture permits a corresponding decentralization of test and validation procedures, with attendant benefits in cost and schedule savings.

2. Performance Features

The most significant performance feature of decentralization in the test and validation task is minimization of system test time in costly facilities with large, expensive test crews.

If the defined degree of system decentralization is implemented and the systemic interactions, through the CMS, are properly and completely defined, fully validated subsystems should operate properly, as an interactive system, when first integrated.

Similarly, as a result of decentralization, absence of a subsystem from the otherwise integrated system should have minimum impact upon the system, and system-level testing should be able to proceed with little perturbation.

3. Flexibility

Flexibility in the test and validation process is a natural outgrowth of its decentralization. This is because of the inherently minimized interaction between subsystems in such a design. The interaction between test and validation processes is correspondingly reduced. Thus temporal coordination between the processes becomes less significant.

Similarly, in a multi-mission environment necessary changes to the operating characteristics and parameter values associated with one subsystem will have minimum effect upon test and validation procedures, other than those associated with that subsystem. Additions to, and deletions from, the space segment configuration in a multi-mission program will have a minimum effect upon test and validation procedures, costs, and schedules.

4. Constraints

TBD

5. Tradeoffs

Tradeoffs between centralized and decentralized test and validation procedures are derived from, and similar to, those associated with centralized and decentralized system design architectures.

The most significant of these is duplication of ancillary equipment capabilities and test procedures, as these are implemented for individual subsystems in the decentralized scheme as opposed to a single, centralized system architecture.

Duplication, on the other hand, inherently provides for parallelism among subsystems in test and validation processes. Thus much of the work, which must inherently be performed in the system test environment when supporting a centralized system, can be completed prior to system integration. The corresponding reduction in the time for system test, and related cost for system test personnel and facilities, appears to be weighted in favor of a decentralized architecture.

6. Other Drivers

TBD

7. Detractors

The principal detractor of the decentralized architecture, in terms of test and validation, is the introduction of inter-subsystem communication and the need for additional activity in support of this function.

At the subsystem level, ancillary equipment must provide simulated communication with other subsystems. Collection and dispersion of information, and the timing of these actions, must be synthesized and/or monitored within the ancillary subsystem test hardware and software. To the extent that these actions are implemented in accordance with complete and accurate interface descriptions, it can be expected that no unforeseen problems will arise in system test of the integrated space segment. Such a presumption cannot, however, serve in lieu of appropriate testing of the composite system.

The general nature of required communication validation can be reasonably defined for a specified decentralized architecture. Specific details will have to be defined uniquely for each implementation of that architecture, including the complement of subsystems and their operating characteristics. Further, changes to the internal software of subsystems to correct problems or fine-tune their performance may affect the sequence, content, or timing of the related communication activities. Test and validation of system communication must be sufficiently comprehensive to determine that all system communication serves the individual needs of the decentralized subsystems when operating together as an integrated system.

8. Conclusions

The primary conclusion that can be drawn from the evaluation of test and validation requirements for centralized vs. decentralized system architectures is that schedule time and related costs for test and validation activities can reasonably be expected to be less for a decentralized system than for a centralized architecture.

This is largely the result of the parallelism possible in test and validation of individual subsystems, prior to system integration, and the reduction in the amount of test and validation associated with individual subsystems which must be performed in the integrated configuration. Additionally, the ability to conduct integrated system-level test and validation activities, despite a missing subsystem, facilitates adherence to schedules and cost minimization in the event of 1) late subsystem delivery to the test and integration activity or 2) a subsystem failure during test and validation.

The fact that a much greater part of the total test and validation activity is shifted to the less costly subsystem environment, in the case of a decentralized design, will further reduce total test and validation costs.

Significantly greater inheritance potential from mission to mission, for the individual subsystems in a decentralized architecture, is inherently applicable to test and validation procedures also, with corresponding long-term savings from project to project.

Additionally, it should be noted that changes, presumably to software, in individual subsystems will generally affect only the test and validation procedures unique to that subsystem, with minimum propagation of these perturbations throughout the integrated procedures executed during system-level testing.

Appendix D

Section D3

A Recommended Hybrid Processing Architecture

I. A RECOMMENDED HYBRID PROCESSING ARCHITECTURE

A. SYSTEM DESCRIPTION

This section defines a hybrid processing architecture, using both centralized and decentralized techniques, to most efficiently satisfy the needs of autonomous generic-class spacecraft having highly sophisticated processing needs. What is proposed is a highly decentralized system architecture similar to that defined in Appendix D2. The primary change is the incorporation of software in the Common Memory Subsystem (CMS) for performance of some system-level executive control functions involving subsystem interactions where arbitrations or protocol decisions must be executed.

A block diagram for the proposed system architecture is given in Figure D3-1. Referring to Figure D3-1, the candidate system consists of seven subsystems which are defined as follows:

1. Common Memory Subsystem (CMS)
2. Telemetry, Tracking, and Command Subsystem (TT&C)
3. Payload Subsystem (P/L)
4. Navigation Subsystem (NAV)
5. Attitude, Translation, and Pointing Subsystem (ATPS)
6. Temperature Control Subsystem (TCS)
7. Power Subsystem (PWR)

Each of the seven subsystems of Figure D3-1 autonomously performs its designated service functions and maintains its own health and welfare. System-level executive control, involving decisions where subsystem interactions are involved, is effected by the CMS. Overall executive control from the ground is provided through the CMS via the spacecraft RF command channel from mission controllers on the ground.

A key concept associated with the architectural design of Figure D3-1 is that the CMS represents the only information transfer interface for all the spacecraft subsystems. Each subsystem (including the CMS) provides a priori-defined internally generated digital information via the Intercom bus to appropriate addresses in the CMS memory for access by other subsystems. Externally generated information required by a subsystem to perform its function can then be accessed by the subsystem through periodic interrogation of appropriate CMS memory addresses. This information may be in the form of spacecraft parametric data from other subsystems, CMS generated executive-level commands, or decoded ground-issued commands transferred to the CMS from the TT&C. It should be noted that the CMS has available, within its own memory, all of the updated information from all subsystems of the spacecraft allowing it to effectively perform its function as the system-level executor.

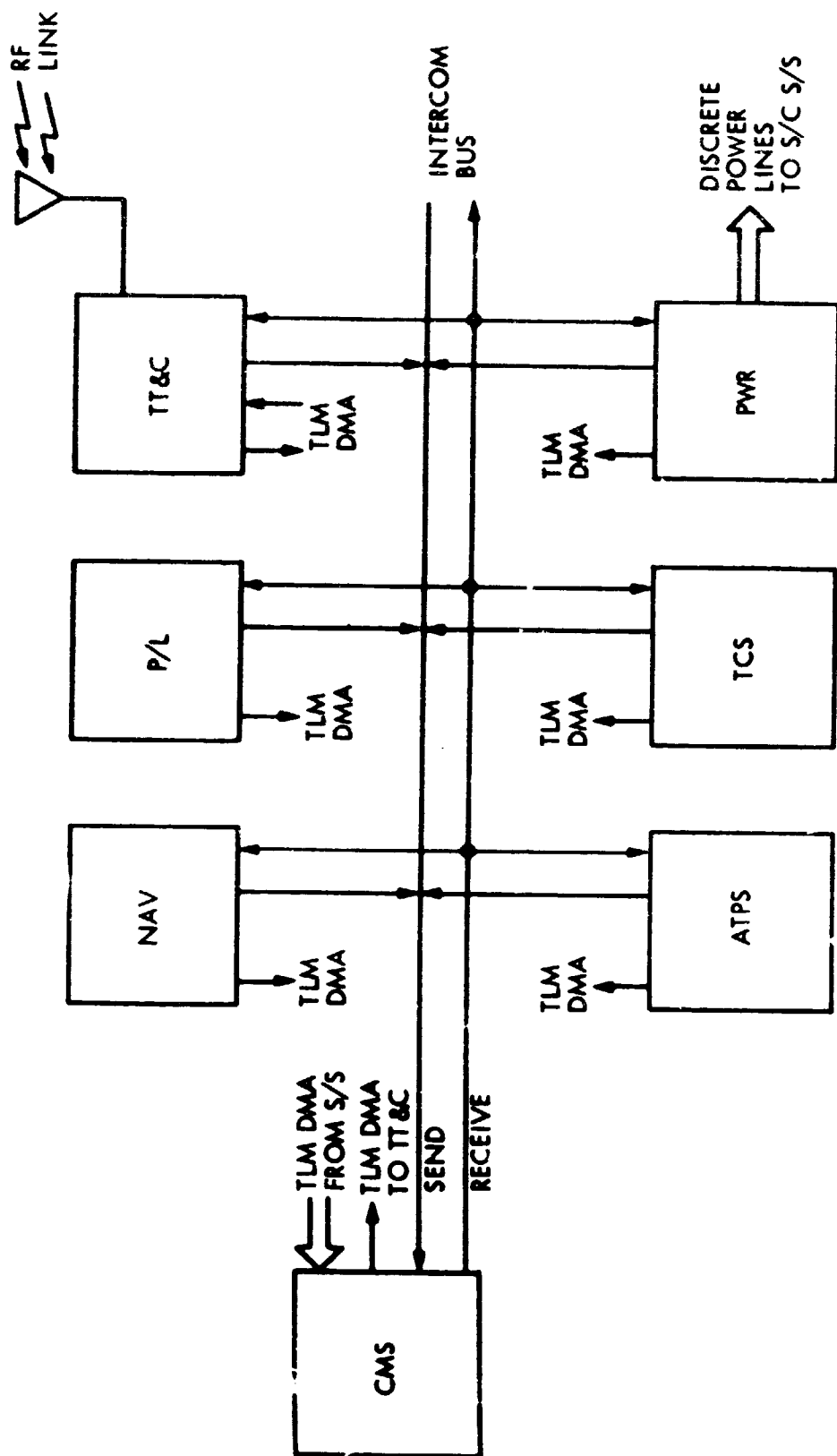


Figure D3-1. System Design Architecture Block Diagram

As part of the system design, time slot allocations are assigned by the CMS to each subsystem for communications with the CMS via the Intercom bus. Such transfer of information from and to any particular subsystem must be accomplished within an allocated time slot for that subsystem. The CMS may reprogram the time slot allocations between subsystems based upon system-level needs.

To minimize Intercom bus traffic needs and maximize information transfer efficiency, telemetry data transfer, which is a continuous demand function, is accomplished independent of the Intercom bus. In the architecture of Figure D3-1, each subsystem acquires, multiplexes, and digitizes its own telemetry information. Referring to Figure D3-1, each subsystem uses a dedicated line to continuously provide its updated digital telemetry data via Direct Memory Access (DMA) to specifically allocated addresses in the CMS. It is also noted that the TT&C accesses, via DMA, the digital telemetry words from these CMS addresses as required to form the output telemetry stream. Since telemetry is a continuous function, this frees the Intercom bus so that it may be solely dedicated to subsystem interactive transfer of nontelemetric information and CMS issued executive-level commands, thereby significantly maximizing information transfer efficiency between subsystems.

It is also noted from Figure D3-1, that power is transferred to each subsystem via dedicated lines from the Power Subsystem (PWR). This is in the form of a single dc voltage level. Therefore, each subsystem must internally generate the specific voltage levels and regulation required for its functional operation.

B. CMS DESCRIPTION

The primary functions of the Common Memory Subsystem (CMS) of Figure D3-1 are 1) to provide an intermediate storage media for information transfer between other spacecraft subsystems, 2) to provide executive-level control of system-level functions involving more than one subsystem, and 3) to distribute a common timing signal to all spacecraft subsystems.

A functional block diagram for a candidate CMS design architecture is given in Figure D3-2. Referring to Figure D3-2, all data to and from all other spacecraft subsystems is routed through the Input/Output (I/O) unit. This includes DMA for telemetry data transfer and Intercom bus traffic for the transfer of interactive information between subsystems and executive-level commands to subsystems.

All spacecraft system-level data and command transfer, with the exception of audit trail readout to the ground, is accomplished through a volatile read/write memory in the Self-Checking Fault-Tolerant Computer block of Figure D3-2. This is the main working memory having high-speed random access capability. As noted in Figure D3-2, there are two additional levels of memory - the nonvolatile buffer memory and the nonvolatile mass memory. The nonvolatile buffer memory interfaces directly with the computer memory, providing a slower access speed but greater capacity than the volatile computer

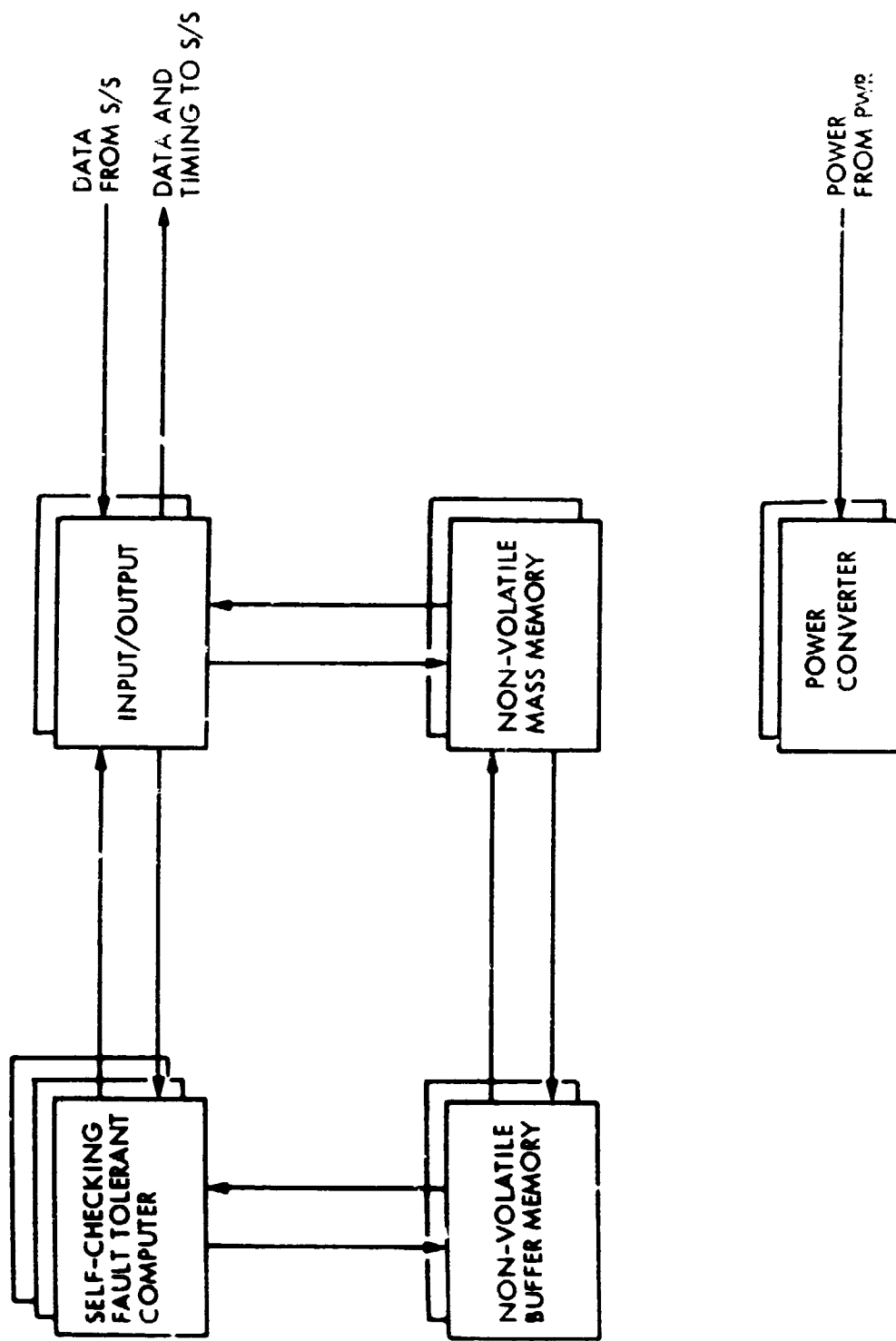


Figure D3-2. C.I.S. Functional Block Diagram

block memory. It stores critical software routines and buffers blocks of audit trail information. The nonvolatile mass memory receives and stores the blocks of audit trail data from the nonvolatile buffer memory. It provides long-term storage of audit trail data for extended periods of autonomous operations. As noted from Figure 2, the nonvolatile mass memory may be accessed directly by the ground through the I/O unit. Also, any data or software stored in either the nonvolatile buffer memory or the nonvolatile mass memory may be accessed by the computer memory for transfer to any spacecraft subsystem as required.

Referring to Figure D3-2, the Self-Checking Fault-Tolerant Computer block provides fault detection, fault isolation, and correction command issuance for not only itself but the remaining blocks of the CMS. This includes the nonvolatile buffer memory, the nonvolatile mass memory, the I/O unit, and the power converter unit, all of which are block redundant.

Typical implementation characteristics might reflect an 8 kiloword to 32 kiloword CMOS computer memory, a 10^6 bit to 10^7 bit bubble memory for the nonvolatile buffer, and a 10^8 bit to 10^9 bit tape recorder for the nonvolatile mass memory.

C. OTHER SUBSYSTEMS

Candidate designs for the remaining subsystems of Figure D3-1, are described in Appendix D2. The hardware designs remain unchanged from the fully decentralized application. They are simply programmed to be responsive to executive-level commands generated and issued by the CMS.

D. EVALUATION

The evaluation of the hybrid system design processing architecture described has been based upon the assumption of mission applications that require sophisticated and complex spacecraft data processing, including the requirement for level-5 or greater autonomy for the entire spacecraft. Furthermore, comparison has been made with fully centralized and fully decentralized architectures only, rather than other hybrid processing architectures.

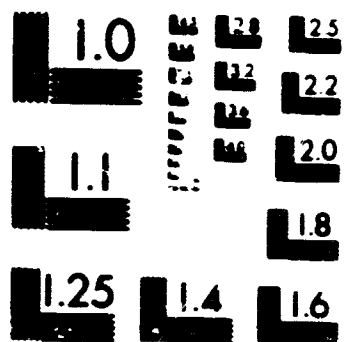
1. Comparison with Fully Centralized Architecture

The hybrid processing architecture described maintains all of the same benefits over a centralized architecture as that provided by the fully decentralized architecture described in Appendix D2. Using the CMS to perform executive-level decisions involving subsystem interactions does not change the system hardware design or degrade the operational capability from that for the fully decentralized system. In fact, executive-level commands generated by the CMS can be transparent, to spacecraft subsystems, from those issued through the CMS by the ground. The potential benefits of the architecture of Figure D3-1 when compared with a fully centralized architecture are summarized as follows:

33

6920

3/B



MICROCOPY RESOLUTION TEST CHART
 NATIONAL BUREAU OF STANDARDS-1963-A

a) Reduced Number and Complexity of Subsystem Interfaces

As more processing is relegated to the subsystem level, less information must be transferred to and from individual subsystems. Therefore, subsystem external interfaces become more simplified. For instance, a typical telemetry subsystem in a centralized architecture would normally require hundreds of analog signal interface lines from subsystems throughout the entire spacecraft. By distributing the processing function, the telemetry operations of analog signal acquisition, multiplexing, and analog-to-digital conversion are accomplished internally by each subsystem for its designated area of measurement responsibility. Therefore, the digitized telemetry measurement information from each subsystem can be provided as a serial stream of bits via a single digital interface line. Referring to the architecture of Figure D3-1, each subsystem, with the exception of the CMS, has a signal interface with only one other subsystem - the CMS. Furthermore, all signal interfaces are digital.

b) Increased Speed and Efficiency of Interactive Information Transfer Between Subsystems

In the hybrid system, telemetry information transfer is handled by means of CMS DMA lines which are independent of the Intercom bus. The Intercom bus is therefore used to transfer only subsystem interactive information (data and commands) between subsystems. Since all noninteractive subsystem processing needs are accomplished internally in each subsystem, the volume of required information to be transferred between subsystems via the Intercom bus, even under crisis conditions, should be grossly reduced when compared to intersubsystem communications required in a centralized processing architecture. Therefore, the speed and efficiency of interactive information transfer between subsystems should be significantly better for the hybrid system architecture.

c) Increased Throughput Rate and Operational Efficiency

Throughput rate and operational efficiency increase in proportion to the number of processing functions that can be performed simultaneously using parallel processors. In a centralized architecture, the processing functions required by each subsystem must be time shared in a single computer. This severely limits the throughput rate and operational efficiency of the system since there is a practical limit in processing capability that is feasible from a single computer based on mass and power considerations. The hybrid architecture of Figure D3-1 dedicates a separate computer to each subsystem. Therefore, the processing requirements for all subsystems may be performed simultaneously. This significantly increases the possible throughput rate and operational efficiency of the system when

compared with a centralized architecture. Furthermore, this overall improvement can still be realized using computer designs that are significantly less complex and demanding in power than those required for a centralized computer implementation.

d) Reduced System Integration Costs

Distribution of most of the processing functions to the subsystem level inherently allows considerable system independence for the test, validation, and operation of subsystems. If the integrity of the system interface requirements for a subsystem is maintained, that subsystem may be almost entirely tested and validated prior to system integration. In contrast, for centralized architectures, because of the comparatively more complex interface requirements and subsystem dependence upon the central processor, very little subsystem test and validation can be accomplished until each subsystem is integrated into the complete system. Therefore, the normally large costs attributed to the spacecraft system integration, test, and operation phases for missions employing centralized system design architectures should be significantly reduced by the use of the hybrid system design architecture of Figure D3-1.

e) Increased Multimission Applicability

An inherent feature of the hybrid processing architecture described in Figure D3-1 is multimission applicability. In contrast, a centralized system design architecture tends to be mission dependent since the performance capability of the central computer has profound effects on the operating limitations of all subsystems. Highly distributed processing, in general, allows considerable system independence for the internal design and operation of subsystems, assuming the integrity of subsystem external interface requirements is maintained. The subsystem signal interface requirements for the architecture of Figure D3-1 involve simply writing into and reading from CMS memory.

Therefore, entire subsystem internal designs could be changed and readily accommodated by the system. Furthermore, old subsystems could be deleted from and new subsystems added to the Intercom bus of Figure D3-1 with no significant perturbations to the overall system hardware design. In like manner, new mission requirements and priorities could be readily accommodated through reallocation of CMS memory space.

f) Increased Growth Potential

The system design architecture of Figure D3-1 inherently provides high growth potential. The internal processing capability of individual subsystems can be significantly increased with little effect on the system design architecture. This is primarily due to the fact that each

subsystem can simultaneously perform its processing functions in parallel with other subsystems. Furthermore, more subsystems can be added to the Intercom bus, limited only by bus traffic capability and CMS memory capacity. Since 1) the Intercom bus traffic involves only interactive information transfer between subsystems and 2) internal subsystem processing greatly reduces the need for such external information transfer, high system throughput rates and operational efficiency characteristics can be realized. In contrast, the growth potential is considerably more limited for a centralized architecture, since the processing and control requirements for all spacecraft subsystems must be accomplished by a single computer on a time-shared basis placing practical limitations on achievable throughput rates and operational efficiencies.

2. COMPARISON WITH FULLY DECENTRALIZED ARCHITECTURE

The hybrid processing architecture of Figure D3-1 is the same as the fully decentralized architecture described in Appendix D2 with the exception that some executive-level control functions, involving interactions between subsystems, are performed by the CMS. This has no effect on the hardware design, being totally implemented in software. Therefore, as noted above, all of the benefits provided by the fully decentralized architecture of Appendix D2, when compared with a fully centralized architecture, are also realized by the hybrid processing architecture of Figure D3-1. However, several potential problems and/or detractors defined for the fully decentralized architecture of Appendix D2, are either resolved or reduced in magnitude by the hybrid approach. These potential benefits of the hybrid architecture of Figure D3-1, when compared with a fully decentralized architecture, are summarized as follows:

a) Increased Information Transfer Efficiency Between Subsystems

Use of the CMS to provide executive control of subsystem intercommunications and system-level decision making should significantly increase the operational efficiency of the Intercom bus as used for the fully decentralized system architecture of Appendix D2. For instance, the CMS could reassign priorities to provide adaptive time slot allocation between subsystems as a function of mission need.

b) Reduced Complexity of Subsystem Fault Routines Associated with System Related Fault Diagnostics and Correction

For the fully decentralized system design architecture described in Appendix D2, the system-level functions of fault detection and correction, where more than one subsystem is involved, must be performed by a set of software fault routines which are distributed among memories of the various subsystems. The subsystem software routine complexities can be significantly reduced by allocating system-level executive responsibility to the CMS for such subsystem interdependent conditions. The

increase in software complexity imposed upon the CMS should be small compared with the total software complexity reduction realized by the subsystems when taken as a whole. This is attributed to the greater efficiency that can be realized by using the CMS for executive control of subsystem interdependent functions.

c) Decreased Criticality of an Early Top-Down System Design

For the fully decentralized system design architecture described in Appendix D2, the system-level executive software responsibilities must be distributed between several subsystem computers. This requires an early top-down system design effort in which the subsystem responsibilities for meeting the system-level needs are properly allocated and well defined. If this is not adequately done, it could potentially impact multiple subsystem software designs later in the program as opposed to one software design if the CMS were used to provide the executive control functions for subsystem interdependent functions. Allocation of appropriate executive control capability to the CMS should remove the need for a system design effort that is any earlier than that required for a fully centralized architecture.

CONCLUSIONS

The hybrid processing architecture described herein retains all of the advantages of the fully decentralized architecture described in Appendix D2 while eliminating most of its negative attributes. A high degree of subsystem test and validation may be accomplished prior to system integration providing the potential for significant mission cost savings compared with a fully centralized architecture. In addition, the proposed architecture is inherently much more adaptable to multimission applications than a fully centralized architecture having comparable performance levels. Furthermore, the system-level software design effort is considerably simplified over that required for a fully decentralized architecture through utilization of a centralized executive-level control capability for functions involving interactions between multiple subsystems. In summary, a highly decentralized architecture, employing centralized executive-level control of functions involving multiple-subsystem interactions, appears to be the most operationally efficient, flexible, and cost effective design approach for future autonomous spacecraft missions that require a high level of processing complexity.

END

FILMED

12-82

DTIC